STOCHASTIC SELF-ASSEMBLY

A Thesis
Presented to
The Academic Faculty

by

Michael J. Fox

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2010

STOCHASTIC SELF-ASSEMBLY

Approved by:

Professor Jeff Shamma, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Magnus Egerstedt
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Martha Grover
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Date Approved: May 2010

To Svetlana

ACKNOWLEDGEMENTS

I would to like to thank Professor Jeff S. Shamma, whose guidance was invaluable to this thesis.

Thesis supported in part by AFOSR projects #FA9550–08–1–0375 and #FA9550–09–1–0538 as well as the National Defense Science & Engineering Graduate Fellowship program.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

We present methods for distributed self-assembly that utilize simple rule-of-thumb control and communication schemes providing probabilistic performance guarantees. These methods represents a staunch departure from existing approaches that require more sophisticated control and communication, but provide deterministic guarantees. In particular, we show that even under severe communication restrictions, any assembly described by an acyclic weighted graph can be assembled with a rule set that is linear in the number of nodes contained in the desired assembly graph. We introduce the concept of stochastic stability to the self-assembly problem and show that stochastic stability of desirable configurations can be exploited to provide probabilistic performance guarantees for the process. Relaxation of the communication restrictions allows simple approaches giving deterministic guarantees. We establish a clear relationship between availability of communication and convergence properties. We consider Self-assembly tasks for the cases of many and few agents as well as large and small assembly goals. We analyze sensitivity of the presented process to communication errors as well as ill-intentioned agents. We discuss convergence rates of the presented process and directions for improving them.

CHAPTER I

INTRODUCTION

It can be argued that the most advanced systems in existence, lifeforms, are constructed in a manner vastly different from most engineered systems. Chemicals are created in extreme quantities (a gram of water contains about 6.022×10^{23} molecules), and at absurd levels of precision, in this manner. These systems self-assemble. Self assembly represents a paradigm shift in the way we build things. Self-assembly is the phenomenon of an ordered structure emerging from the aggregate behavior of simpler constituent entities acting autonomously. Despite the obvious scalability, reliability, and, parallelizability advantages inherent in self-assembly, the development of a theory of these dynamic processes is nascent. Interest in self-assembly has increased in recent years though, alongside the general spike in the investigation of distributed systems.

There are two streams of work in self-assembly, molecular and robotic. The molecular self-assembly field is not expressly concerned with programmable dynamic system models and is noted for its importance to the application of self-assembly theory. The robotics field utilizes self-assembly in several different capacities. Sometimes, a robot is designed in a modular fashion so that it can accomplish a wider variety of objectives by reconfiguring its constituent parts. In other formulations, a machine utilizes a form of self-assembly to automatically repair itself or generate locomotion.

In any event, few have considered self-assembly in a theoretical context, most preferring heuristic approaches to specific applications. Several theoretical formulations have been advanced in recent years. These approaches have led to analytical results where a wide range of different assemblies can be constructed. In some cases there is some global

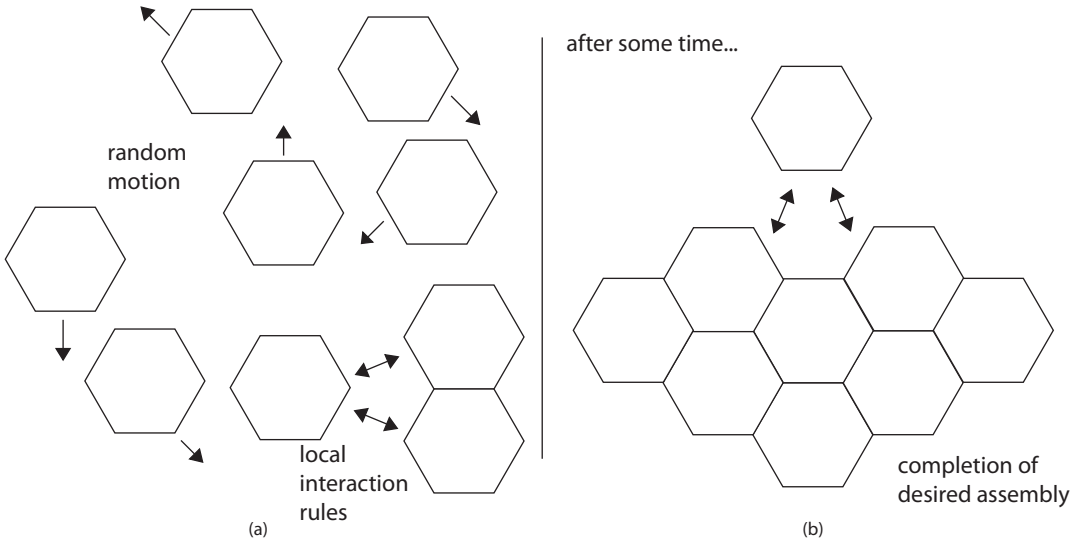

Figure 1: (a) The parts move about randomly, forming structures according to local interaction rules. (b) A specific arrangement of the parts is desired in the long-run.

information embedded in the process and in others there is extensive communication between the agents. In either case, the form of the performance guarantee is deterministic. A specific, desirable result can be expected. The rule set employed by the agents can be very complicated though and does not scale well with the complexity of the assembly desired.

We propose an approach to self-assembly where simple rules can be utilized to arrive at probabilistic performance guarantees. This approach represents an important advancement to the theory for several reasons:

- Many chemical self-assembly processes feature dynamic equilibria. Desirable assemblies that are themselves stochastic processes may be acceptable or even desirable in some situations

- Self-assembly is often relevant to processes involving huge numbers of simplistic agents. Achieving a high yield of desired assemblies will often be preferable when a marginally higher yield requires substantial increases in the capabilities of the individual agents.

- Establishing benchmark performance results using very simple, generic, and easily

synthesizable control schemes provides insights into the more powerful convergence results. How does agent memory, computational capability, sensor sophistication, and communication capacity translate into better self-assembly performance?

Our work is the first to insist on simple rules and allow probabilistic guarantees. In particular, we introduce stochastic stability, an equilibrium concept from the game theory and stochastic systems literature, as a useful tool for understanding dynamic systems whose performance is related to a yield of desired sub-structures. Figure 1 illustrates our concept.

The import of developing an abstract theory of self-assembly can be understood by analogy to integrated circuits. Modern integrated circuits can feature hundreds of millions of transistors. Each of these devices is in and of itself a technological marvel. Precise manufacturing tolerances must be met for each one of these elements to function properly. The behavior of these devices is described by semiconductor physics. However, a software developer or hardware engineer can use this technology with little or no understanding of carrier drift or density of states. The wealth of clear and appropriate abstractions in integrated circuits and computing at every hierarchical level allows tasks to be accomplished with intuitive thinking.

Chemical interactions are governed by thermodynamics. Those who understand how to arrive at complex structures through laboratory chemistry are often not the same people who understand why certain structures are desirable for medical, industrial, or other purposes. Dialog and collaboration between these different disciplines can be improved through the development of a useful hierarchical decomposition of the relevant processes.

We focus on a relatively high level of abstraction. We consider individual agents, or parts, which may represent molecules, macromolecules, or robots. We make simplifying assumptions about the behavior of these agents that will be more appropriate to some disciplines than they will be to others. We strive to be as general as possible, but acknowledge that there will be applications areas that will not correspond to our framework.

We will concern ourselves exclusively with scenarios where the agents' general motion is stochastic. Our analysis will actually abstract away the agents' motion and will be valid whenever certain bounds on the probability of two agents coming into proximity can be assumed. An agent's action in response to stimuli, described by its rules of operation, represent perturbations to the agent's otherwise random behavior. This conception of the dynamics as essentially stochastic with limited actuation can be seen as analogous to chemical assembly processes. In that context, the motion of the particles can be ascribed to thermal motion or the motion (as by mixing) of the solution. Assembly is achieved via the long-run statistical propensity for certain bonds to be created or severed. This decreased number of degrees of freedom available for control design for these processes is an essential characteristic. This phenomenon has some intersect with the study of under-actuated systems. Stochastic self-assembly can be thought of as an entirely different discipline from deterministic self-assembly procedures, often relevant to robotics. In robotics it is often possible or even necessary for the constituent robot parts to behave in very predictable ways throughout the process— something we will not be concerned with.

It should be noted that it is difficult to conceive of any immediate limit to the capabilities of self-assembling systems. Human beings self-assemble from simplistic embryos floating in raw materials. However, despite the lofty possibilities, the field is yet to produce a defining "killer app". As such, there is nothing resembling a definitive framework for validating models of self-assembly processes. We therefore urge the reader to understand that in what follows, we consider our primary contribution to be the identification of stochastic stability as a relevant and useful concept in the analysis of self-assembling systems. We will propose models that illustrate how desirable yields in a stochastic self-assembly process can be related to stochastically stable states. We provide justifications for our system models, but in no way assert that the relevance of stochastic stability is limited to the models we propose. In the same vein, the techniques we utilize in providing rigorous mathematical proof of our claims can be modified to establish similar results for different models of the

underlying processes.

The remainder of this thesis is organized as follows: In chapter 2 we survey the existing work in stochastic self-assembly. In chapter 3 we propose a self-assembly process that relies on severe assumptions about information available to each sub-assembly. In chapter 4 we provide a brief review of stochastic stability. In chapter 5 we utilize the ideas of chapter 4 in the presentation of a self-assembly process that does not depend on the aforementioned assumptions. In chapter 6 we explore several extensions to the process presented in chapter 5 and then chapter 7 concludes.

CHAPTER II

PREVIOUS WORK

2.1 History

Since almost all chemical and biological systems self-assemble, there is no specific origin of the scientific investigation of self-assembly. However, we will henceforth consider self-assembly as referring to processes with the following two features:

- Lack of a regular pattern
- A unique terminal structure

These two qualities differentiate the study of certain forms of crystallization and polymerization from the study of self-assembly as presented in this work. While the term self-assembly is certainly applicable even when these qualities are absent, we will henceforth restrict ourselves to these type of problems.

The first to utilize mathematics to describe these processes in abstract terms was John Von Neumman, who saw tremendous potential for technological advancement through creating machines that could replicate themselves. Von Neumann first envisioned a kinematic self-replicating machine . In this model, a complete machine moves about a "sea" of parts and builds a copy of itself according to preloaded instructions. It then copies the instructions onto the new machine, and it in turn repeats the process. Von Neumann later developed a more abstract model using cellular automata (CA). A cellular automaton is a system in which rules are applied to cells in a regular grid. Von Neumann originally proposed his ideas in 1948 [1], [2], but they were not widely disseminated until they appeared in a 1955 issue of Scientific American [3].

Von Neumman's results have had some applicability limitations, due in no small part to the state of the art of computing in the 1950's. Very much like Richard Feynman's now

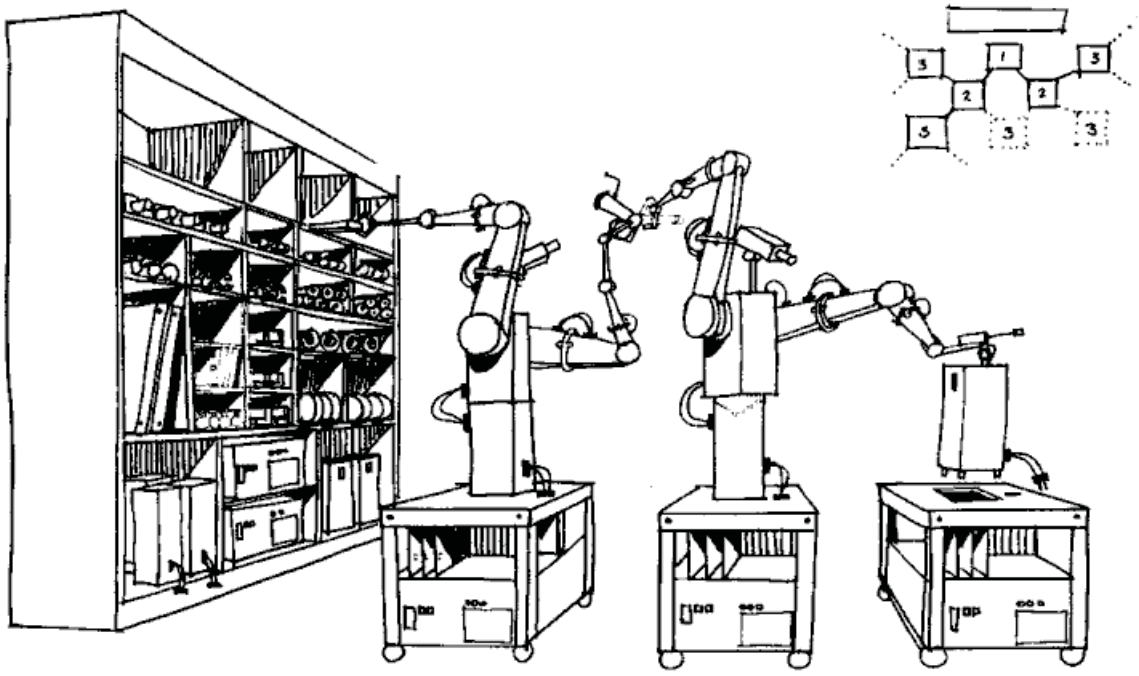

Figure 2: A 1980's cartoon of a self-replicating machine [7].

famous lecture that is often credited with ushering in the interest in nanotechnology, Von Neumman is frequently credited with bringing attention to the area of self-assembly, but his work is almost entirely of historical interest in the present day. Usage of notation consistent with CA can be found in [4], but the algorithms for synthesizing decision procedures are novel. CA is also used in [5] to model DNA-based self-assembly. Von Neumman's existence results in some ways created and destroyed at once the interest in abstract modeling and analysis for self-assembly. Von Neumman's tessellation model was a 29 state per cell CA and he was able to show that it could make endless copies of itself. He termed it a *universal constructor*.

John Conway's two-dimensional, two-state CA called Game of Life [6] became popular in the 1970's. Life is a simple CA that exhibits a very wide range of behaviors. A favorite pastime of curious mathematicians— and a notorious consumer of after-hours CPU time, Life ignited interest in the possibility of emergence and self-organization allowing for simple processes to create sophisticated results. Many patterns have been studied in Life

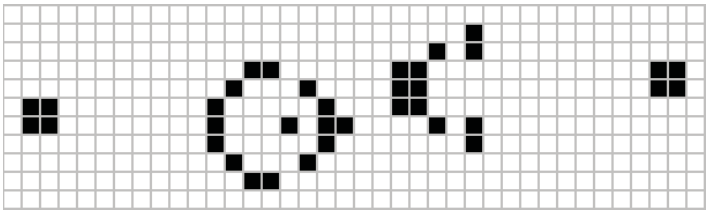

Figure 3: The glider gun in Game of Life is able to generate moving gliders perpetually.

including gliders (simple perpetual motion machines), guns (patterns producing gliders), and breeders (patterns producing guns). It was soon realized that these patterns could be arranged to realize combinational logic elements and even a Turing complete computer. Figure 3 (available: http://en.wikipedia.org/wiki/Conway's_Game_of_Life) shows a popular form in "Life", the glider gun. We recommend visiting the webpage provided for the image to observe the fully animated representation of the glider gun's behavior.

Von Neumman also discussed probabilistic self-assembly although it was not very well-defined. This approach involves stochastic state transitions by the agents so that the phenomenon of evolution is potentially exploitable. This method has not received much attention prior to this work. We will consider stochastic behavior of the agents (learning). However, the learning is implemented as noise, or perturbation of an underlying deterministic process that does not change over time. In actual biological evolution, the agents' underlying rule-set changes over time allowing the cells to self-assemble while in the long-run simultaneously mutating and undergoing natural selection. Investigation of the latter process requires a clearly defined environmental influence as well as a dynamic process over a space of dynamic processes.

In the 1970's, robotic manipulators in machining centers utilized a common interface for multiple attachable appendages. In the late 1980's [8] applied this idea to the entire robot. Modular robotics is an active area of research. Generic algorithms [4], [10], [11] relevant to modular robots as well as some robots [9], [12], [11] have been developed. Some modular robots [12] can actually operate independently, but can combine to achieve

better modes of locomotion. Stochastic state transition is becoming more relevant to this branch of self-assembly as the scale becomes smaller and the number of modules in a robot grows. The desire for reliable, high-performance robot functionality has typically kept this field focused on achieving quality performance with a small number of robots operating and switching between a small number of configurations of the modules. In this context, the need for stochastic behavior of the individual robotic agents has been minimal.

2.2 Related work

In recent years a theory of self-assembly has been advanced [13] utilizing a form of stochastic behavior that is not unlike our own approach. The agent motion is random and only the formation and severance of bonds is controllable. Many different problems in this area have been studied including algorithms for estimation of parameters over a time-varying communication network [14] (consensus) as well as optimal distributions of products of a severance decision [13]. The theory has been demonstrated with a programmable parts testbed (an air-table with modular robots). The problem of synthesizing control policies for general assemblies has been addressed [15], but the results have relied on exhaustive search and storage of all possible sub-assemblies of a desired assembly and require communication and consensus algorithms. Alternatively, in [16] a simpler procedure is presented, but it is assumed that infinitely many agents are present. To circumvent issues arising in the case of finitely many agents, [16] suggests augmenting the process with additional communication in the form of intra-assembly consensus algorithms. Our work addresses these obstacles. We seek a simplistic method of synthesizing control procedures for any assembly from a very general class. We find that a dramatic reduction in the complexity of agent communication and decision making can be realized if we allow for probabilistic performance guarantees.

Agent complexity is an important feature for biomolecular and other microscale self-assembly processes that utilize agents whose communication, computation, and memory

capabilities are not easily augmented. A prime example are DNA tiles [5]. These devices have been shown to be capable of performing simple computations such as XOR and therefore can produce simple forms using those computations, e.g. Sierpinski triangles. At this scale and below, themes such as agent communication and knowledge must be understood figuratively. In this setting, the agent's state and the rules governing its behavior are purely thermodynamic. The agents have none of the memory, processing, or motor capabilities of modular robots. The usage of anthropomorphic language is intended in a general and figurative sense. That said, in order to enable substantial advance in self-assembly it is advantageous to continually lower the complexity that theoretical models assume to be available when creating algorithms.

Self-assembly is in fact a very broad field that encompasses many different techniques, scales, and approaches. We have made certain important distinctions related to our work: stochastic, theoretical, low-complexity, and probabilistic guarantees. Another classification is that of formal vs. functional self-assembly. We focus on formal self-assembly. That is, we desire the outcome of the parts arranged in a specific geometric arrangement (sometimes described abstractly by a weighted graph). Other approaches have sought assemblies accomplishing specified functions as opposed to forms. Functional assemblies such as self-healing materials and passive electronic components have been studied [17]. This work also has the distinction of implementation alongside existing top-down manufacturing processes such as photo-lithography. Our work, as is common in formal self assembly, utilizes a one-pot approach. This is desirable for any such synthesis, as separation and purification can be costly and time consuming.

CHAPTER III

ASSEMBLY-LEVEL INFORMATION PROCESS

3.1 Atoms and assemblies

Throughout this thesis, we will introduce control policies that, when applied to a system model, result in a dynamic process with desirable asymptotic behavior. That is, we desire that these processes reach and remain in a desirable state. In all processes considered, we will have N identical agents, or atoms. These atoms should be understood as abstractions that represent two- or three-dimensional geometric shapes. We will illustrate examples using specific geometric definitions, but the results will be presented in a general manner applicable to a very wide range of atom types. The state of the system will always be the arrangement of these atoms. In some circumstances the atoms will store additional information in the form of internal states that may not be functions of the physical state of the system. This information will influence the future decisions of the agents and therefore the future physical state of the system. Performance goals will always be related to the physical state of the system alone. The procedures for updating the internal states of the atoms will be available to the control designer.

Furthermore, the state of the system corresponds to the spatial arrangement of the parts and can be thought of as a non-unique representation. This is because the labeling of the parts will be arbitrary due to their being identical atoms. We will not be interested in any absolute coordinates of the parts, only their orientation with respect to neighboring parts. We will understand two parts as adjacent (in contact) whenever they are connected (share a bond), but adjacent parts are not necessarily bonded. A set of parts such that there exists a path of connected parts between any two parts is called an assembly. A single atom can be thought of as an assembly of size one. Our performance goal will always be to maximize

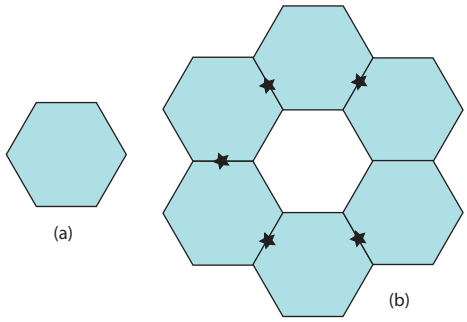

Figure 4: (a) A part is an assembly of size one. (b) An assembly of size six, with stars indicating the bonds. Notice that there is a pair of edges that are adjacent, but do not share a bond.

the number of a given assembly present. Figure 4 illustrates these ideas with hexagonal parts in the plane.

We will explore stochastic self-assembly from the perspective of the assembly and the atom separately. The difference is one of information. Operating from the standpoint of the assembly assumes that the atoms participating in an assembly can share information accurately and act in unison. This severe assumption allows a simple algorithm to converge to the maximum number of desirable assemblies almost surely. In addition, we can demonstrate this result using very simple notation since we need not represent the role of each atom in our description of the state of the system, rather we need only indicate the concentrations of the various assembly types at each time instant. The results that follow are not extremely novel given that a similar approach has been taken by [13] with explicit intra-assembly communication procedures for the atoms to coordinate their actions. What is novel however is that we abstract away the communication problem and show that when a suitable communication procedure can be assumed, the assembly procedure can be accomplished with minimal difficulty. The procedure is quite general and immediately applies to a very broad class of atom and assembly types. This format is inspired by group-based decision processes in the game theory literature. In that context, at each time instant, several players select an action from a group action set. Since this specific scenario is simpler than the general case of group-based decision (as their are no utility functions to speak of), we

do not introduce that concept formally or use related notation.

Additionally, treating the case of self-assembly with severe communication assumptions will highlight the impact of dropping these assumptions in the next chapter. In particular, we will see a much weaker convergence result that is not obviously improvable without significant augmentation of agent communication capabilities.

The problem and solution are presented in abstract terms in the next four sections. We provide an example in the last section of this chapter.

3.2 Problem formulation

3.2.1 Preliminaries

There are N atoms of one fixed type of polygon or polyhedron. Two atoms can form to create a new assembly. Two assemblies can form to create a still larger assembly. An assembly made of two or more atoms can break up back into atoms. We will not be concerned with breakups into assemblies larger than individual atoms. There are P possible assembly types forming a set $\mathcal{P} = \{1, 2, 3, ..., P\}$. The atoms are assigned to 1 as a convention. The size of a polygon $p \in \mathcal{P}$ denoted by size(p) is defined as the number of atoms in p. Clearly $\forall\, p \in \mathcal{P}, \text{size}(p) \leq N$.

Let $\triangle$ be the space of probability distributions over $\mathcal{P}$. A *reaction* is a map

$$R : \{(i, j) \in \mathcal{P} \times \mathcal{P} : \text{size}(i) + \text{size}(j) \leq N\} \rightarrow \triangle.$$

If $i, j \in \mathcal{P}$ and $f = R(i, j)$ then the set $\{p \in \mathcal{P} : f(p) \neq 0\}$ are called the *products* of i and j. The map R specifies the probability of each possible product when two atoms interact and has the following properties

$$\begin{cases} \text{if } p \text{ is a product of } i \text{ and } j \text{ then } \text{size}(p) = \text{size}(i) + \text{size}(j) \\ R(i, j) = R(j, i)\ \forall\, i, j \in \mathcal{P}. \end{cases}$$

These properties can be understood as conservation of mass and the irrelevance of the order of the reactants, respectively.

3.2.2 System model

The state at time t, $S(t)$, is a P-vector where the ith element corresponds to the number of assemblies of type i at time t. The state $S(t)$ has the property that $\forall t$, $\sum_{i \in \mathcal{P}} S_i(t) \cdot \text{size}(i) = N$. The state space is denoted by $\mathcal{S}$.

At each time t, a random variable $F(S(t), t)$ is sampled, taking on values in the set $\mathcal{C}(S(t)) = \{i \in \mathcal{P} : S_i(t) \neq 0\} \times (\{j \in \mathcal{P} : S_j(t) \geq \chi_i(j) + 1\} \cup \{0\})$, where $\chi_i(\cdot)$ is the indicator function for i. Each $(i, j) \in \mathcal{C}(S(t))$ provides a present assembly type i and either an additional assembly type j, or $j = 0$ for i by itself. We can have $i = j$ only when at least two of that assembly type are present. If $j \neq 0$ then one of the actions from the set $\mathcal{A} = \{\texttt{form}, \texttt{break.}i, \texttt{break.}j, \texttt{null}\}$ is taken.

`form`:

$$\begin{cases} S_i(t+1) = S_i(t) - 1 \\ S_j(t+1) = S_j(t) - 1 \\ k = \text{rand}(R(i,j)) \\ S_k(t+1) = S_k(t) + 1 \\ S_l(t+1) = S_l(t),\ \forall\, l \neq \{i, j, k\} \end{cases}$$

`break.`i:

$$\begin{cases} S_i(t+1) = S_i(t) - 1 \\ S_1(t+1) = S_1(t) + \text{size}(i) \\ S_l(t+1) = S_l(t),\ \forall\, l \neq \{i, 1\} \end{cases}$$

`break.`j:

$$\begin{cases} S_j(t+1) = S_j(t) - 1 \\ S_1(t+1) = S_1(t) + \text{size}(i) \\ S_l(t+1) = S_l(t),\ \forall\, l \neq \{j, 1\} \end{cases}$$

`null`:

$$S(t) = S(t-1)$$

Where rand(·) is the result of sampling a random variable with the specified distribution. If $j = 0$ then i is selected and the available actions are `break.` i and `null` only. Usage of the verbs form and break is understood throughout to mean selection of assemblies (or agents in the atom-level process) and undertaking of the appropriate actions.

3.2.3 Assembly task

There is a polygonal assembly $G \in \mathcal{P}$ that is the desired, or *goal*, assembly. The objective is to bring the state of the system to the set $\bar{S} = \{S \in \mathcal{S} : \sum_{i \neq G} S_i(t) \cdot \text{size}(i) < \text{size}(G)\}$ and stay there. An assembly task is *feasible* if for any initial time t_0, and any initial state $S(t_0)$ with $S_1(t_0) \geq \text{size}(G)$ there exists a time $T > t_0$ and a sequence of assembly selections $i(t), j(t)$ and corresponding actions $a(t)$ in $[t_0, T-1]$ that result in $S_G(T) > S_G(t_0)$ with positive probability. Note that feasibility depends on the specifics of G, R, and F.

What remains is to establish policies for the agents so that they can decide on a particular action based on the information available to them at the time of the decision. The policy we will present will require the agents in the two assemblies to be aware only of the number of atoms in each assembly and whether either assembly is of type G.

3.3 Policies

3.3.1 Information restricted policy

An *information restricted policy*, hereon just a *policy*, is a pair of mappings $H = \{H_1, H_2\}$. The map $H_2 : \mathcal{P} \times \mathcal{P} \to \mathcal{A}$ satisfies $\forall\, i, j, k, l \in \mathcal{P}$ such that

$$\begin{cases} \text{size}(i) = \text{size}(k) \\ \text{size}(j) = \text{size}(l) \\ \chi_G(i) = \chi_G(k) \\ \chi_G(j) = \chi_G(l) \end{cases}$$

we have $H_2(i, j) = H_2(k, l)$. In other words, H_2 can depend only on the function size(·) and whether or not i and/or j are equal to G. The map $H_1 : \mathcal{P} \to \mathcal{A}$ satisfies $\forall\, i, k \in \mathcal{P}$

such that $\chi_G(i) = \chi_G(k)$ we have $H_1(i) = H_1(k)$, so that H_1 depends only on whether or not $i = G$ At time t, H_1 is used to select an action if $\sum_{i \in \mathcal{P}} S_i(t) = 1$ and H_2 is used to select an action when $\sum_{i \in \mathcal{P}} S_i(t) \geq 2$.

3.3.2 Goal-size policy

At time t with $F(S(t), t) = (i, j)$, $\hat{H}_1$ is used to select an action if $j = 0$ and $\hat{H}_2$ is used to select an action when $j \neq 0$. Assume without loss of generality throughout that if two assemblies, i and $j \neq 0$, are selected at time t, size$(i) \leq$ size(j). Let the *goal-size* policy $\hat{H} = \{\hat{H}_1, \hat{H}_2\}$ be defined

$$\hat{H}_1 = \begin{cases} \texttt{break } i, & i \neq G \\ \texttt{null}, & i = G \end{cases}$$

$$\hat{H}_2 = \begin{cases} \texttt{form}, & \text{size}(i) < \text{size}(G) \text{ and size}(j) < \text{size}(G) \\ \texttt{break.}j, & \text{size}(j) \geq \text{size}(G) \text{ and } j \neq G \\ \texttt{null}, & j = G \end{cases}$$

The correctness of this policy will rely on the following definition.

Definition 2.1 (F bounded away from zero): *There exists $\bar{F} > 0$ such that for all $t, S(t)$*

$$\mathbf{Pr}\left[F(S(t), t) = (i, j)\right] > \bar{F}$$

for all $(i, j) \in \mathcal{C}(S(t))$.

The distribution F bounded away from zero guarantees that two assemblies present at time t have a probability of being selected that is bounded away from zero. This removes the possibility of pieces becoming shielded and inaccessible such as from containment within a larger cyclical assembly. The next two sections will provide the proof of the correctness of $\hat{H}$— that it brings the system to $\bar{S}$ and stays there.

3.4 Monotonicity of S_G

This section will build up to the result that S_G strictly increases with positive probability under $\hat{H}$. We will first note an elementary result concerning S_G.

Claim 3.1 (S_G non-decreasing under $\hat{H}$):

$$S_G(t) \geq S_G(t-1) \,\forall\, t$$

Proof: The only ways that S_G can decrease are (i) a goal assembly being selected and broken up or (ii) a goal assembly forming with another assembly. $\hat{H}$ never does either of these things. ■

This implies that $\bar{S}$ is always an invariant set under $\hat{H}$. In practice, there are many different sequences of actions that can lead to an increase in S_G. However, to ease the analysis, we will restrict our interest to one specific sequence of actions— one consisting only of *form* actions.

Claim 3.2 (*form*-only trajectories): *For any feasible assembly task $G \neq 1$ with F bounded away from zero, for any initial time t_0, and any initial state $S(t_0) \notin \bar{S}, S_1(t_0) \geq$ size(G), there exists a time $T > t_0$ and a sequence of assembly selections $i(t), j(t)$ in $[t_0, T-1]$ that, along with the* `form` *action at every time instant, result in $S_G(T) = S_G(t_0) + 1$ with positive probability.*

Proof: By induction on size(G); size$(G) = 2$: Any assembly of size 2 can only be formed from two atoms. Since these two atoms are available at time t_0 we can select them and have $S_G(t_0 + 1) = S_G(t_0) + 1$.

Induction step: Let the claim hold for all G with size$(G) < k$ and let size$(G) = k$, then G can only be formed from assemblies whose sizes sum to k, but this implies that each of these assemblies has size less than k and can therefore be reached via a `form`-only trajectory implying that G itself can be reached with only `form` actions. ■

Theorem 3.1 (S_G increases with positive probability under $\hat{H}$): *For any feasible assembly task G with F bounded away from zero, if $S(t_0) \notin \bar{S}$, then there exists a time $T > t_0$*

that results in $S_G(T) > S_G(t_0)$ under $\hat{H}$ with positive probability.

Proof: Case 1: Suppose $G = 1$, then we are guaranteed to eventually select one or two assemblies with size greater than one with positive probability (by F bounded away from zero). $\hat{H}$ will call for the break up one of those assemblies, increasing S_G.

Case 2: Suppose $G \neq 1$ and $S_1(t_0) \geq \text{size}(G)$, by the feasibility of the assembly task G and the previous claim there exists such a trajectory involving `form` actions only. It follows from the details of the proof of that claim that the sequence of assembly selections, $i(t), j(t)$, can be chosen so that $\text{size}(i(t)) < \text{size}(G)$ and $\text{size}(j(t)) < \text{size}(G)$ for all $t \in [t_0, T-1]$. Since $\hat{H}$ always chooses `form` when the assemblies both have size less than $\text{size}(G)$, and there is positive probability associated with $i(t)$ and $j(t)$ (by F bounded away from zero), S_G increases with positive probability under $\hat{H}$.

Case 3: Suppose $G \neq 1$ and $S_1(t_0) < \text{size}(G)$, if we have for some $j \neq G$ that $S_j(t_0) > 0$ and $\text{size}(j) \geq \text{size}(G)$ then the action `break.`j will be taken with positive probability under $\hat{H}$, bringing us to the previous case. If not, then $\hat{H}_2$ will, with positive probability, form smaller assemblies until an assembly j satisfying $S_j(t) > 0$ and $\text{size}(j) \geq \text{size}(G)$ for some t is formed. If $j = G$ we are done, if not then `break.`j is eventually performed with positive probability, giving the previous case.

The next section completes the proof of the correctness of $\hat{H}$ with respect to the assembly task.

3.5 Correctness of $\hat{H}$

Claim 3.3 (S_G increases almost surely under $\hat{H}$): *For any feasible assembly task G with F bounded away from zero, If $S(t_0) \notin \bar{S}$, then*

$$\forall\, \delta > 0,\ \exists\, T \text{ such that } \mathbf{Pr}\left[S_G(T) > S_G(t_0)\right] > 1 - \delta$$

Proof: Fix $M = S_G(t_0)$ and let

$$\mathcal{S}_{G,M} = \{S \in \mathcal{S} : S_G = M\}$$

By the previous claim, for each state in $\mathcal{S}_{G,M}$ there is a positive probability γ that S_G will increase within τ iterations. Let $\hat{\gamma}$ be the minimum of these probabilities, and $\hat{\tau}$ the maximum of these times. The maximum and minimum are always defined because $\mathcal{S}_{G,M}$ is always a finite set. We have

$$\mathbf{Pr}\left[S_G(t_0+\hat{\tau}) > S_G(t_0)\right] > \hat{\gamma}$$

$$\mathbf{Pr}\left[S_G(t_0+2\hat{\tau}) > S_G(t_0)\right] > 1-(1-\hat{\gamma})^2$$

$$\vdots$$

$$\mathbf{Pr}\left[S_G(t_0+k\hat{\tau}) > S_G(t_0)\right] > 1-(1-\hat{\gamma})^k.$$

Choose k so that $(1-\hat{\gamma})^k < \delta$, then $T = k\hat{\tau}$. ■

We can now assert the efficacy of $\hat{H}$.

Theorem 3.2 (Correctness of $\hat{H}$): *Consider a feasible assembly task G with F bounded away from zero. From any initial time and state $t_0, S(t_0)$, $\hat{H}$ achieves the assembly task of reaching and staying in $\bar{S}$ almost surely.*

Proof: The set $\bar{S}$ is invariant under $\hat{H}$, so what must be shown is for $S(t_0) \notin \bar{S}$

$$\forall\, \delta > 0,\ \exists\, T > t_0 \text{ such that } \mathbf{Pr}\left[S_G(T) \in \bar{S}\right] > 1-\delta$$

Let $S_G(t_0) = m$, let $\lfloor \frac{N}{\text{size}(G)} \rfloor = M$, and let $\hat{\delta} = (1-\delta)^{\frac{1}{M-m-1}}$. By the previous claim there are times $\tau_m, \tau_{m+1}, ..., \tau_{M-1}$ such that

$$\mathbf{Pr}\left[S_G(t_0+\tau_m) > m\right] > \hat{\delta}$$

$$\mathbf{Pr}\left[S_G(t_0+\tau_m+\tau_{m+1}) > m+1\right] > \hat{\delta}^2$$

$$\vdots$$

$$\mathbf{Pr}\left[S_G(t_0+\tau_m+\tau_{m+1}+...+\tau_{M-1}) = M\right] > \hat{\delta}^{M-m-1} = 1-\delta$$

so $T = \sum_{k=m}^{M-1} \tau_k$. ■

We remark that analyzing an assembly task only requires knowledge of the reactions on the set $\{(i,j) \in \mathcal{P} \times \mathcal{P} : \text{size}(i) + \text{size}(j) < \text{size}(G)\}$ because larger assemblies that are formed can only be broken up afterwards. The last section of this chapter provides an example of the assembly-level information process using equilateral triangles in the plane. The details of the procedure are illustrated and a MATLAB simulation is conducted.

3.6 Example: Equilateral triangles in the plane

The advantage of the self-assembly process presented using assembly-level information (the *goal-size* policy) is that it achieves a very strong convergence result using very simple rules. If we can assume the existence of reasonably reliable communication between a connected network of agents (an assembly) and a suitable algorithm for achieving consensus among the agents about the needed information, then the number of desired assemblies will almost surely increase monotonically, achieve the maximum, and remain there. We do not address the specific details of implementing such a consensus algorithm. We consider the problem to be related to k-consensus (see [18]) and other problems related to consensus for discrete values over graphs. It is not necessary to even identify the exact form of the different states the system can take. In fact, these are the ingredients needed for the results of this chapter to apply:

- The agents must be identical.
- The agents must form into a finite set of assemblies.
- The desired assembly must be *feasible* per the definition.
- The information and communication requirements must be met so that we can utilize the abstraction of assembly-level actions.

A simple illustration of the procedure can be arrived at using equilateral triangles. This choice has the added benefit of having been previously used in [13], so its use here provides an easy contrast for the reader considering both approaches.

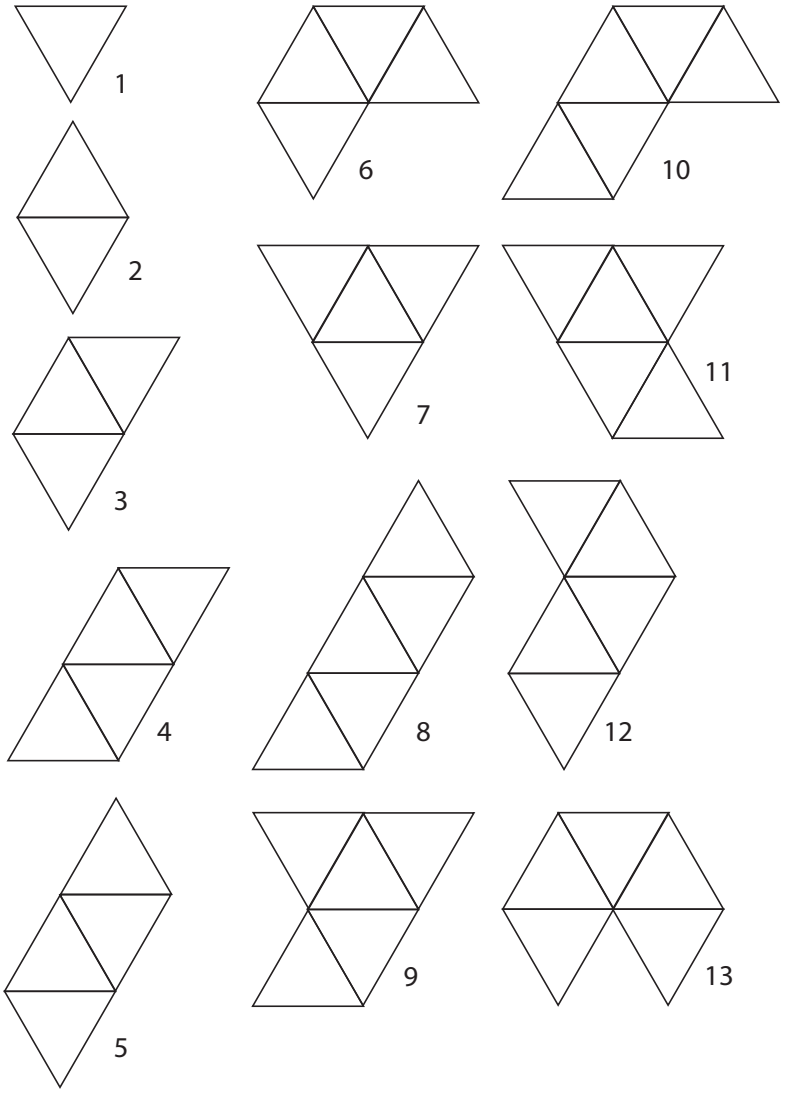

Figure 5: There are 13 assemblies obtainable by joining five or fewer equilateral triangles along their sides.

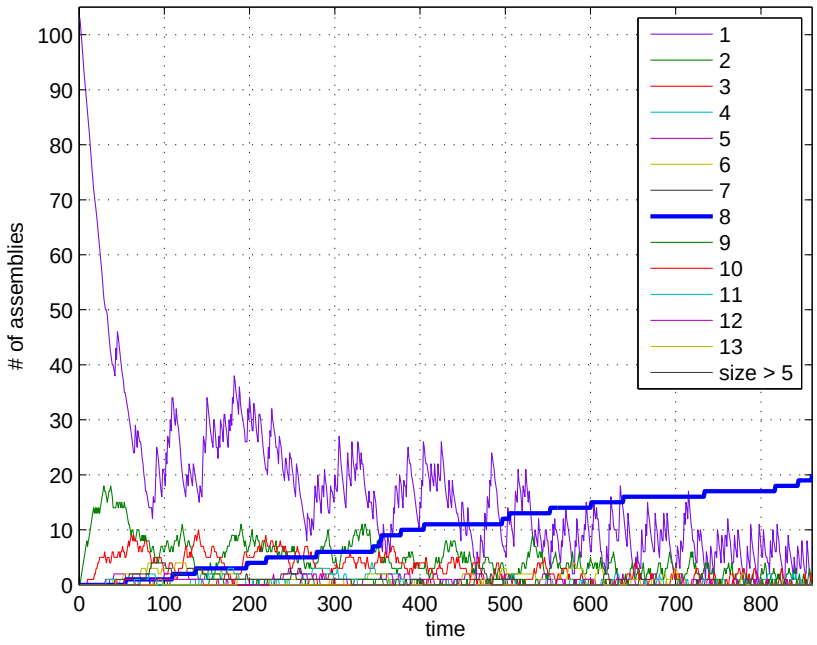

Figure 6: The simulation reaches the desired state of twenty assemblies of type 8 from an initial condition of 103 assemblies of type 1.

Figure 5 shows the different assemblies containing up to five atoms. In this case the atoms are equilateral triangles and the larger assemblies are formed by overlapping a single side. In our simulation, we seek to generate as many assemblies of type 8 as possible. Since type 8 includes five atoms, it is necessary to tabulate the reactions involving product assemblies of size up to and including five atoms. A reaction is a set of possible product types when two assembly types form together. Reactions with larger products do not need to be determined explicitly because assemblies that contain more atoms than the desired assembly will eventually be broken up into type 1 assemblies in the current dynamic process. The reactions for our case are shown in the table below and are easily verifiable.

Clearly the number of atoms is conserved in every reaction. In our simulation we begin from the trivial initial condition of all parts having no connections— type 1. The convergence result is actually applicable from every initial condition. Figure 6 shows the results of the MATLAB simulation.

Reactant 1	Reactant 2	Product(s)
1	1	$\{2\}$
1	2	$\{3\}$
1	3	$\{4, 5, 6, 7\}$
1	4	$\{8, 9, 10\}$
1	5	$\{8, 11, 12\}$
1	6	$\{9, 10, 11, 12, 13\}$
1	7	$\{9, 11\}$
2	2	$\{4, 5, 6\}$
2	3	$\{8, 9, 10, 11, 12, 13\}$

The rest of this thesis will be concerned with self-assembly techniques and analysis when the assumption of assembly-level information is dropped and only atom-level information is available. That is, an atom will have to make a decision with limited knowledge about the assembly it is participating in. The convergence result, weaker than that found in this chapter, will be in the form of stochastic stability. The next chapter will introduce this concept.

CHAPTER IV

STOCHASTIC STABILITY

This introduction to the notion of Stochastic Stability will draw heavily from the presentation of Young [19]. The interested reader may want to consult that source because we will develop these concepts here with an eye for brevity. For an even more succinct exposition, consult [20].

The well-known notion of dynamic system stability refers to the properties of the equilibrium points of a system. If an autonomous system is in an equilibrium state, it will stay there indefinitely. Stability classifies the attractiveness of these equilibrium states. Different stability concepts (local, global, asymptotic, exponential, etc.) describe the behavior of the system at various proximities to a particular equilibria. In the next chapter we will present a model of self-assembly that can be viewed as a Markov process with fixed transition probabilities that are then perturbed with small noise terms. Under the unperturbed model, there are a great many equilibrium states, the vast majority of which are undesirable. However, we will see that when we add the stochastic perturbations in a particular way, the desirable states can be made to dominate the long-run statistical behavior of the system. In particular, the desirable states are stochastically stable. Stochastic stability of a set of states means that those states are observed with non-vanishing probability as a temperature parameter ϵ approaches zero.

4.1 Regular perturbed Markov processes

We will consider a Markov process P^0 on a finite state space Z. We will restrict our interest to perturbations to this process of a specific form, defined below.

Definition 4.1 (Regular perturbed Markov process): *Let P^ϵ be a Markov process on Z for each $\epsilon \in (0, \bar{\epsilon}]$. P^ϵ is a regular perturbed Markov process if P^ϵ is irreducible (we can*

get to any state from any state in finite time with positive probability) for every $\epsilon \in (0, \bar{\epsilon}]$ *and for each* $z, z' \in Z$ *we have*

$$\lim_{\epsilon \to 0} P^{\epsilon}_{zz'} = P^{0}_{zz'},$$

and

$$\textit{if } P^{\epsilon}_{zz'} > 0 \textit{ for some } \epsilon > 0, \textit{ then } 0 < \lim_{\epsilon \to 0} P^{\epsilon}_{zz'} / \epsilon^{r(z,z')} < \infty \textit{ for some } r(z, z') \geq 0.$$

The quantity $r(z, z') \in \mathbb{R}$ is called the *resistance* of the transition $z \to z'$. Clearly, $r(z, z')$ must be uniquely defined in order to satisfy the condition. Also, $P^{0}_{zz'} > 0$ if and only if $r(z, z') = 0$. That is, transitions that occur with non-zero probability under P^0 have zero resistance. Transitions that never occur can be considered as having infinite resistance so that $r(z, z')$ is always defined.

For each ϵ, there is a unique stationary distribution, μ^{ϵ}, associated with P^{ϵ} (by its irreducibility). We can now formally define stochastic stability.

Definition 4.2 (Stochastic stability): *A state* z *is stochastically stable (Young, 1993) if*

$$\lim_{\epsilon \to 0} \mu^{\epsilon}(z) > 0.$$

It has been shown elsewhere that the above limit exists for every z so that every regular perturbed Markov process has at least one stochastically stable state. These states are the ones that the system spends most time in over the long run when ϵ is small. Now we will describe how to compute the stochastically stable states.

4.2 Computing the stochastically stable states

In this section we will explain how to compute the stochastically stable states and provide an illustrative example. It should be noted that the stochastically stable states correspond to the perturbed process P^{ϵ}. That is, which states survive in the presence of the perturbations will depend on how the perturbation is introduced. It is possible to arrive at different

stochastically stable states for the same process P^0 by applying the perturbations differently. Also, the stochastically stable states correspond to the limiting case of ϵ approaching zero, and are not always particularly likely to be observed when ϵ is not small.

4.2.1 The resistance tree method

A recurrent class of a Markov process is a set of states that, from any state in the set one can reach any other state in the set in finite time with positive probability, and no state outside the set is accessible from any state inside it. Let P^0 have K recurrent classes $E_1, E_2, ..., E_K$. We will define for every distinct pair of recurrent classes E_i and E_j, $i \neq j$, a sequence of states $\zeta = (z_1, z_2, ..., z_q), z_1 \in E_i, z_q \in E_j$ called an *ij-path*. The resistance of the path is the sum of resistances in the sequence, $r(\zeta) = r(z_1, z_2) + r(z_2, z_3) + ... + r(z_{q-1}, z_q)$. We further denote $r_{ij} = \min r(\zeta)$ as the ij-path with least resistance. r_{ij} is always positive because there cannot be a zero resistance path between two distinct recurrent classes.

Now, for each recurrent class E_j, construct a tree rooted at a vertex j corresponding to E_j. That is, a set of $K-1$ directed edges such that each $E_i, i \neq j$ is represented by a vertex i and there is a unique directed path from any vertex different from j to j. The resistance of such a tree is the sum of the resistances r_{ij} on the $K-1$ edges. The stochastic potential γ_j of the recurrent class E_j is the minimum resistance among all such trees rooted at j. We expect the recurrent classes of minimum stochastic potential to be the most likely when ϵ is small. This result has been formalized (Young, 1993) as follows:

Theorem 4.1: *Let P^ϵ be a regular perturbed Markov process, and let μ^ϵ be the unique stationary distribution of P^ϵ for each $\epsilon > 0$. Then $\lim_{\epsilon \to 0} \mu^\epsilon = \mu^0$ exists, and μ^0 is a stationary distribution of P^0. The stochastically stable states are precisely those states that are contained in the recurrent class(es) of P^0 having minimum stochastic potential.*

We will now consider an example that is simple, but sufficient to illustrate the procedure.

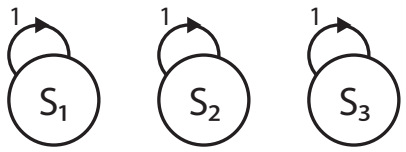

Figure 7: A simple three-state Markov process, P^0. The system stays in its initial state for all t.

4.2.2 Example: A three-state process

Figure 7 illustrates a simple three state Markov process, P^0. Each state $S_i, i \in \{1, 2, 3\}$ is a singleton recurrent class, also referred to as an *absorbing* state. The state transition matrix is given by

$$P^0 = I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

It is easy to see that all state distributions are stationary in this example because the transition matrix is the identity matrix. This process is not particularly interesting, until we introduce the perturbation and arrive at P^ϵ as shown in Figure 8. It can easily be verified that P^ϵ satisfies Definition 4.1, so it must have at least one stochastically stable state. In this example, we can easily see that the minimum resistance rooted trees for each state (as each is a recurrent class of P^0) must be

$$S_1 \to S_3 \to S_2, r = 2 + 1 = 3$$

$$S_3 \to S_2 \to S_1, r = 1 + 1 = 2$$

$$S_2 \to S_1 \to S_3, r = 1 + 2 = 3$$

where the resistance is also noted. All other rooted trees will involve crossing states more than once, so that all other trees rooted at S_i will have a single edge equal to the resistance above in addition to other non-zero resistance edges. We can conclude in this example that there is one stochastically stable state, S_1. This result is intuitively satisfying because among the three states, the probability of leaving S_1, ϵ^2, is least.

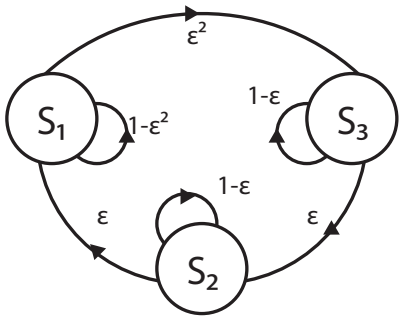

Figure 8: The perturbed process P^ϵ has $r(S_1, S_2) = 2, r(S_3, S_2) = 1, r(S_2, S_1) = 1$.

In cases with more edges in P^ϵ and more states, finding the minimum resistance tree rooted at S_i will be more difficult, although it is always a finite search. We do note, however, that the search applies only to recurrent classes of states and these are oftentimes much fewer in number than the states themselves.

CHAPTER V

ATOM-LEVEL INFORMATION PROCESS

We are now prepared to revisit the problem of self-assembly with agent information limited to the atom-level. That is, agents are not necessarily aware of the entire structure of the assembly that they take part in. In particular, two agents are selected at each time t and they are able to communicate to each other only their internal states and the internal states of agents that they share edges with. We note that this does not imply a 'two-hop' communication. Since agents communicate their own internal states during a formation or severance event, their 'full' internal state can be considered to be their own internal states as well as those of their neighbors. The agents must decide upon an action to undertake from a set of available actions based only on the information noted above. The agents will be able to form or sever edges or do nothing. They will also be able to update their internal states. All agents besides the two selected randomly at time t cannot form or sever edges and cannot change their internal states. This rather severe limitation makes it impossible for all agents to have complete assembly-level information. This is because if two agents form an edge, one agent will never be apprised of additional edges formed or severed by the other agent afterwards. This limitation is particularly relevant to molecular self-assembly processes where more elaborate communication may not be realizable. In any case, we will see that even under these limitations there exist decision policies that will converge to desirable states.

5.1 System model

There are N identical parts, or *atoms*, in the plane. The desired, or *goal* assembly, is a connected weighted graph, $\hat{G} = \{\hat{\mathcal{N}}, \hat{E}, \hat{W}\}$, where $\hat{\mathcal{N}} = \{1, 2, ..., \hat{N}\}$ are nodes, $\hat{E} \subset \hat{\mathcal{N}} \times \hat{\mathcal{N}}$ are undirected edges, and $\hat{W} : \hat{E} \to \mathcal{W}$ are edge weights from a finite set $\mathcal{W}$, representing

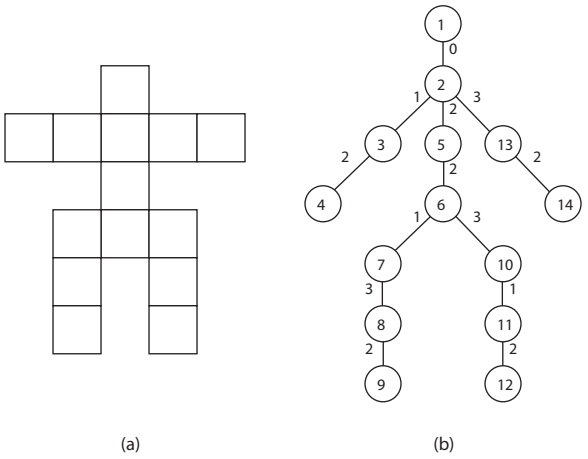

Figure 9: (a) An assembly involving squares in the plane, and (b) its representation as a weighted graph.

the orientation of the connection. The system is described by the physical state, a graph $G = \{\mathcal{N}, E, W\}$, along with the internal state vector S, where $\mathcal{N} = \{1, 2, ..., N\}$ are nodes, $E \subset \mathcal{N} \times \mathcal{N}$ are undirected edges, $W : E \to \mathcal{W}$ are edge weights, and $S \in \hat{\mathcal{N}}^N$ is a state vector with the ith element representing the internal state information of node i. The internal state will be used to store each nodes' corresponding vertex in $\hat{G}$— its role in the assembly. We will assume throughout that $\hat{G}$ is acyclic— a tree. We also impose on $\hat{G}$ that vertex 1 participates in only one edge, $(1, 2)$, with $W(1, 2) = 0$. Since the nodes are atoms and $\hat{G}$ is a tree this imposition does not sacrifice any generality.

Figure 9 illustrates our model using the goal assembly that will serve as a motivating example throughout our discussion. In this case, the atoms are squares in the plane. Edges are realized as fully overlapping sides. We can consider each node other than 1 as having a path of directed edges from node 1 terminating at it because $\hat{G}$ is a tree. Facing this edge, the other edges at a node are assigned weights according to their orientation— 1 for nodes to the left, 2 for nodes behind, and 3 for nodes to the right. Any acyclic assembly composed of squares in this manner can be represented using this method. There are several ways to represent the same assembly, but any graph corresponds to only one assembly. We could have chosen the starting edge (given weight 0) differently, which would have given a

different graph, however, it would still describe an assembly of squares unambiguously.

5.2 Statement of objectives

The objective of this work is to construct a dynamic process that obeys the guidelines suggested at the start of the chapter and converges to a state that is desirable with respect to the goal assembly. In order to define this objective clearly, we will need to review some concepts in graph theory.

5.2.1 Graph isomorphism and subgraphs

We say that two graphs are an *isomorphism*, or one graph is *isomorphic* to another when they obey an equivalence relation. That is $G_1 = \{\mathcal{N}_1, E_1, W_1\} \simeq \{\mathcal{N}_2, E_2, W_2\} = G_2$ if $\exists f : \mathcal{N}_1 \to \mathcal{N}_2$ such that

$$\begin{cases} (i,j) \in E_1 \text{ iff } (f(i), f(j)) \in E_2 \\ W_1(i,j) = W_2(f(i), f(j)) \,\forall\, (i,j) \in E_1 \end{cases}$$

An equivalence class of graphs all represent the same assembly. Since it is self-assembly performance that we are concerned with, our objective will be phrased in terms of equivalence classes of graphs. We will no longer explicitly refer to these equivalence classes. We will simply refer to graphs and assume that the reference to the equivalence class is understood.

Given $I \subset \mathcal{N}$ we define the *subgraph* $G \cap I = \{\mathcal{N} \cap I, E \cap I \times I, W|_{I \times I}\}$. We say that G contains H if a subgraph of G is isomorphic to H. A connected subgraph is maximal if there are no nodes in the original graph that could have been added to the subgraph while still leaving it connected. We will use the terms assembly and maximal connected subgraph interchangeably.

5.2.2 The objective

The performance objective can now be stated as follows: We seek to maximize the number of disjoint subgraphs of G that are isomorphic to $\hat{G}$. If N is not an integral multiple of $\hat{N}$

then it will not be possible for all atoms to be participating in goal assemblies at any one time. We do not stipulate any preference for the behavior of these remainder agents. It will turn out that these nodes will always be part of connected subgraphs that are isomorphic to subgraphs of $\hat{G}$ in the process that will be presented. Using the simple rules that we will suggest, the dynamic process will not always converge to the maximum number of desirable assemblies present. However, the number of agents not participating in goal assemblies will be bounded. This way, by making N large we can get as close to 100 percent of all nodes participating in goal assemblies as we want.

5.3 The self-assembly process

This section will introduce the self-assembly process for atom-level information, the most important contribution of this work. The process will be a regular perturbed Markov process. Therefore, we will present the streamlined unperturbed process first and then introduce the perturbations that are necessary for our results. The unperturbed process is capable of producing goal assemblies and will even produce the maximum number of goal assemblies with positive probability from some initial conditions. However, the unperturbed process will usually land in some absorbing state that is not particularly desirable and never leave.

The process we present may not appear to be the most intuitive way to to construct desired assemblies. In particular, we restrict ourselves to assembling one piece at a time. We do not ever connect two assemblies without at least one having only one vertex. This serial assembly process will nevertheless be sufficient to achieve our performance guarantees for any acyclic $\hat{G}$

5.3.1 The unperturbed dynamic process (P^0)

Let $G(t) = \{\mathcal{N}, E(t), W(t)\}, S(t)$ be the physical and internal states, respectively, of the system, and let $n_{G(t)}(i)$ represent the neighbors of node i. At time t, two agents, i and j, $i \neq j$, are selected according to a random process $F(G(t), t)$. Assume w.l.o.g. that

$|n_{G(t)}(i)| \leq |n_{G(t)}(j)|$ and if $|n_{G(t)}(i)| = |n_{G(t)}(j)|$ then $S_i(t) \leq S_j(t)$. One of the four actions from the set $\mathcal{A} = \{\texttt{initial form}, \texttt{form}, \texttt{break}, \texttt{null}\}$ is taken

`initial form`: If $|n_{G(t)}(i)| = |n_{G(t)}(j)| = 0$, set

$$\begin{cases} E(t+1) = E(t) + (i,j) \\ (W(t+1))(k,l) = \begin{cases} 0 & (k,l) = (i,j) \\ (W(t))(k,l) & (k,l) \neq (i,j) \end{cases} \\ S_k(t+1) = \begin{cases} S_k(t) & k \notin \{i,j\} \\ 1 & k = i \\ 2 & k = j \end{cases} \end{cases}$$

`form`: If $|n_{G(t)}(i)| = 0$, $|n_{G(t)}(j)| \neq 0$, and there exists $s \in \hat{\mathcal{N}}$ and $w \in \mathcal{W}$ such that $\hat{W}(S_j(t), s) = w$ and for all $k \in n_{G(t)}(j)$ we have $S_k(t) \neq s$ (if there are several such pairs (s, w), choose one at random uniformly) set

$$\begin{cases} E(t+1) = E(t) + (i,j) \\ (W(t+1))(k,l) = \begin{cases} w & (k,l) = (i,j) \\ (W(t))(k,l) & (k,l) \neq (i,j) \end{cases} \\ S_k(t+1) = \begin{cases} S_k(t) & k \neq i \\ s & k = i \end{cases} \end{cases}$$

`null`: Otherwise, set

$$\begin{cases} E(t+1) = E(t) \\ W(t+1) = W(t) \\ S(t+1) = S(t) \end{cases}$$

The `initial form` action creates a new assembly when two free atoms are selected. The `form` action checks for a vacancy when a free atom and an atom participating in an

assembly are selected. When neither of these actions are available for the pair of nodes selected at time t, `null` is defaulted to.

5.3.2 Absorbing states of P^0:

Consider P^0 with $E(0) = W(0) = \{\emptyset\}$ and (by convention) $S(0) = 0^N$. We will concern ourselves with $N \geq \hat{N}$ for obvious reasons.

Claim 5.1 (Absorbing states of P^0): *The absorbing states of P^0 are as follows:*

(i) If $N \neq m\hat{N} + 1$ for any positive integers m then the absorbing states are all states where each connected subgraph of G is isomorphic to a subgraph of $\hat{G}$ and $|n_G(i)| \neq 0$ for all $i \in \mathcal{N}$.

(ii) If $N = m\hat{N} + 1$ for some positive integer m then the absorbing states are all states where each connected subgraph of G is isomorphic to a subgraph of $\hat{G}$. Either $n_G(i) \neq 0$ for all $i \in \mathcal{N}$ or there exists only one $i \in \mathcal{N}$ with $|n_G(i)| = 0$ and all other nodes are part of connected subgraphs isomorphic to $\hat{G}$ itself.

Proof: (i) Since $|n_G(i)| \neq 0$ for all $i \in \mathcal{N}$, the `form` and `initial form` actions cannot be taken and the state of the system cannot change. If there exists i such that $|n_G(i)| = 0$ then either there is a j such that $n_G(j) = 0$ (so that i and j can `initial form`) or there is a k such that there exists $s \in \hat{\mathcal{N}}$ and $w \in \mathcal{W}$ such that $\hat{W}(S_k(t), s) = w$ and for all $l \in n_{G(t)}(k)$ we have $S_l(t) \neq s$

(ii) If $|n_G(i)| \neq 0$ for all $i \in \mathcal{N}$ then the state cannot change. Otherwise there can be no `initial form` because there is only one i satisfying $|n_G(i)| = 0$. Also, there can be no `form` because for all $j \in \mathcal{N}$, $s \in \hat{\mathcal{N}}$, and $w \in \mathcal{W}$ such that $\hat{W}(S_j(t), s) = w$ we have some $k \in n_{G(t)}(j)$ such that $S_k(t) = s$. The argument for all other states not absorbing is similar to the above. ■

In words, the above claim states that the absorbing states cannot form any further and have all assemblies being sub-assemblies of the desired assembly. Only in the situation described in (ii) is it possible to have an atom without any connections in an absorbing

state, and even it cannot form any further.

Note that while there are states satisfying the objective among the absorbing states, not all absorbing states satisfy the objective. These undesirable absorbing states are also reached with non-zero probability in P^0. The claim establishes the recurrent classes of P^ϵ as precisely these absorbing states, which themselves may be thought of as fully built states. That is, no more *form* actions can be undertaken. The next subsection introduces the perturbed process, P^ϵ.

5.3.3 Perturbed dynamic process (P^ϵ)

P^ϵ is the same as P^0 except that null is subdivided as follows:

`break`: If $(i,j) \in E(t)$ and $n_{G(t)}(j) \backslash i = \{\emptyset\}$ then case 1: $S_i(t) = 1, S_j(t) = 2$ and we set, with probability ϵ

$$\begin{cases} E(t+1) = E(t) - (i,j) \\ S_k(t+1) = \begin{cases} S_k(t) & k \notin \{i,j\} \\ 1 & k \in \{i,j\} \end{cases} \end{cases}$$

or case 2: $S_i(t) \neq 1$ and we set, with probability ϵ

$$\begin{cases} E(t+1) = E(t) - (i,j) \\ S_k(t+1) = \begin{cases} S_k(t) & k \neq j \\ 1 & k = j \end{cases} \end{cases}$$

and in either case we set, with probability $1 - \epsilon$

$$\begin{cases} E(t+1) = E(t) \\ S(t+1) = S(t) \end{cases}$$

Otherwise perform `null` as above.

The `break` action severs an edge with probability ϵ when two connected nodes are selected with one node being an extremity (having only one edge) of the assembly. The

exception is nodes with state 1 which are not broken off unless there is only one edge in its maximal connected subgraph. The unperturbed process is simply the above with $\epsilon = 0$.

The next section will develop convergence results for P^ϵ. P^ϵ allows the severance of an edge where one node is yet to establish an additional edge. In other words, a node that is an extremity of the assembly, when selected, breaks off with probability ϵ.

5.4 Stochastically stable states of P^ϵ

The results of this section will depend upon the following assumption:

Assumption 5.1 (F bounded away from zero): There exists $\bar{F} > 0$ such that for all $t, G(t)$ we have

$$\mathbf{Pr}\left[F(G(t), t) = (i, j)\right] > \bar{F} \,\forall\, (i, j) \in \{(i, j) \in \mathcal{N} \times \mathcal{N} : i \neq j\}.$$

This assumption is analogous to Assumption 3.1 and can also be interpreted as the inability for any of the parts to become isolated from further reactions. For our purposes, it suffices to note that since F is bounded away from zero it can be neglected in our analysis of stochastic stability because it does not contribute to resistance.

The stochastically stable states of P^ϵ will form a class defined by the number of distinct maximal connected subgraphs. In other words, the stochastically stable states of P^ϵ are all of the absorbing, or fully built states with a specific number of assemblies. In particular, it is the absorbing states with the minimum number of assemblies that are stochastically stable.

Claim 5.2 (Minimum assembly count): *The stochastically stables states of P^ϵ are the absorbing states with the minimum number of maximal connected subgraphs with disjoint vertices. In particular, there are $\lceil N/\hat{N} \rceil$ such subgraphs in all of the stochastically stable states.*

Proof: The proof is based on the construction of rooted trees for the claimed class of stochastically stable states and comparison with the trees corresponding to all other

absorbing states. Let Z_0 be the absorbing states of P^ϵ. We will decompose Z_0 into disjoint sets Z_m where each state in Z_m has m assemblies, so that

$$\bigcup_{m \in \mathcal{M}} Z_m = Z_0, \mathcal{M} = \{\lceil N/\hat{N} \rceil, \lceil N/\hat{N} \rceil + 1, ..., \lfloor N/2 \rfloor\}.$$

The rooted trees for each absorbing state contain $|Z_0| - 1$ edges in each tree. There is nonzero resistance associated with each of these edges because the states are all absorbing. For P^ϵ, the resistance is at least one for each edge. We will show that a rooted tree satisfying this minimum resistance of $|Z_0| - 1$ can be constructed for each state in $Z_{\lceil N/\hat{N} \rceil}$.

To shorten the proof we restrict our interest to the case where both N and $\hat{N}$ are even. A similar construction exists for the cases where either N, $\hat{N}$, or both are odd. Let $z_{N/2} \in Z_{N/2}$ be the state with all assemblies as pairs of nodes. Let $z_{N/2-1} \in Z_{N/2-1}$ be the state arrived at by performing `break` on one of the pairs and then performing `form` actions to attach the two free atoms to one of the other pairs (or two if $\hat{N} = 3$). We proceed like this for each $z_m \in Z_m$ letting z_{m+1} be arrived at by breaking up a pair and transferring the pieces to the largest possible assembly with a `break` action followed by two `form` actions. Figure 10, 11 illustrates an example of this procedure.

We will first construct the tree rooted at $z_{\lceil N/\hat{N} \rceil}$. There are edges corresponding to the z_m as follows:

$$z_{\lfloor N/2 \rfloor} \rightarrow z_{\lfloor N/2 \rfloor - 1} \rightarrow ... \rightarrow z_{\lceil N/\hat{N} \rceil}.$$

Each edge $z_m \rightarrow z_{m-1}$ represents breaking up one assembly of two nodes ($r = 1$), and then having those nodes form together. This will form the backbone of the tree. Figure 12 illustrates the concept we will use to complete the tree rooted at $z_{\lceil N/\hat{N} \rceil}$.

Each row contains the states with a particular number of assemblies. What remains to be shown is that all states in Z_m can reach z_m via a path through states in Z_m with all edges having resistance 1. Consider a state $y \in Z_m, y \neq z_m$. Each z_m consists of only two-node assemblies, completed assemblies, and at most one other assembly. Let x_m refer to the largest incomplete assembly of z_m (let it be a two-node assembly if that is the largest). We

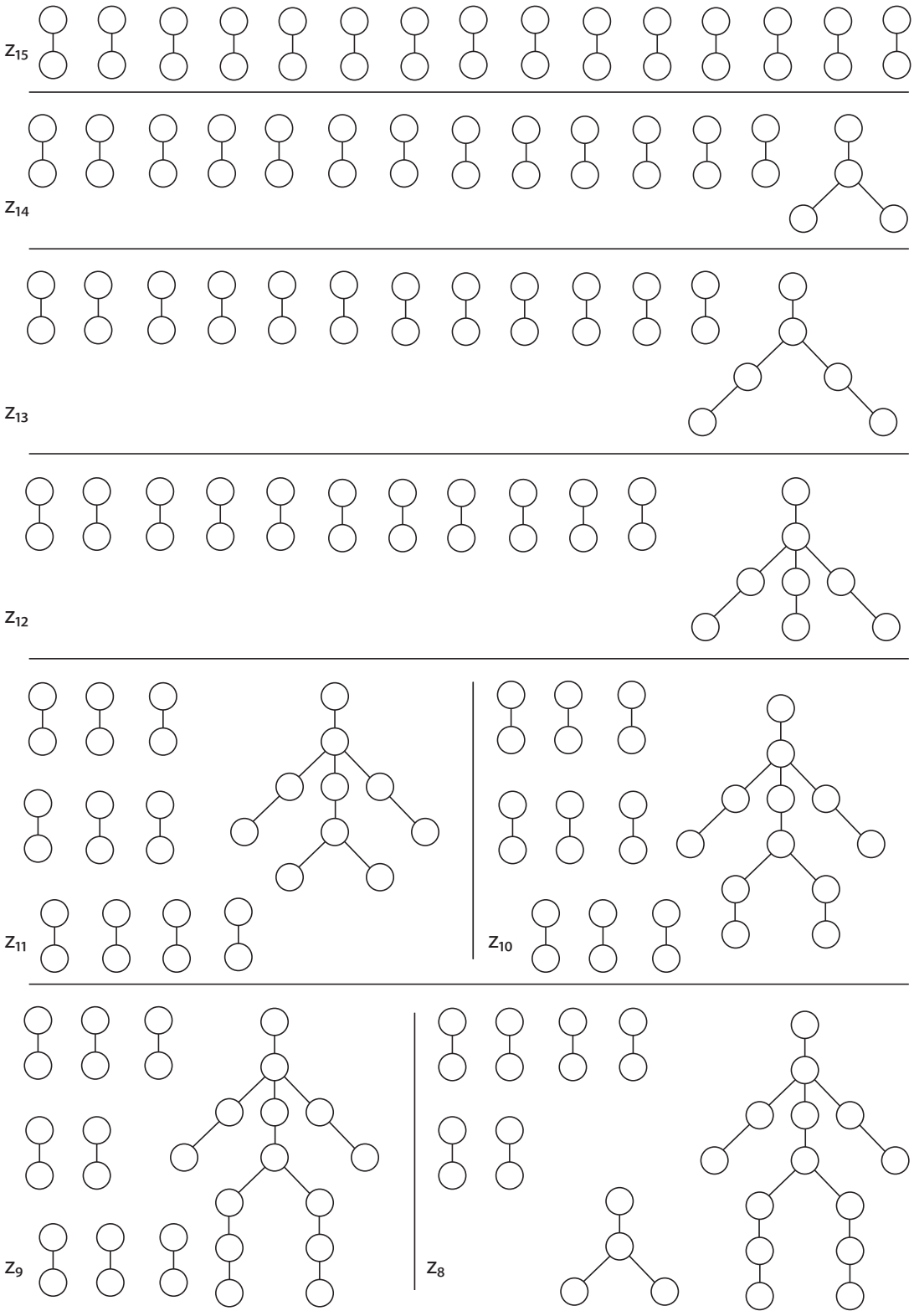

Figure 10: The states $z_m, m \in \{8, 9, ..., 15\}$ for the $\hat{G}$ introduced above, $N = 30$, $\hat{N} = 14$.

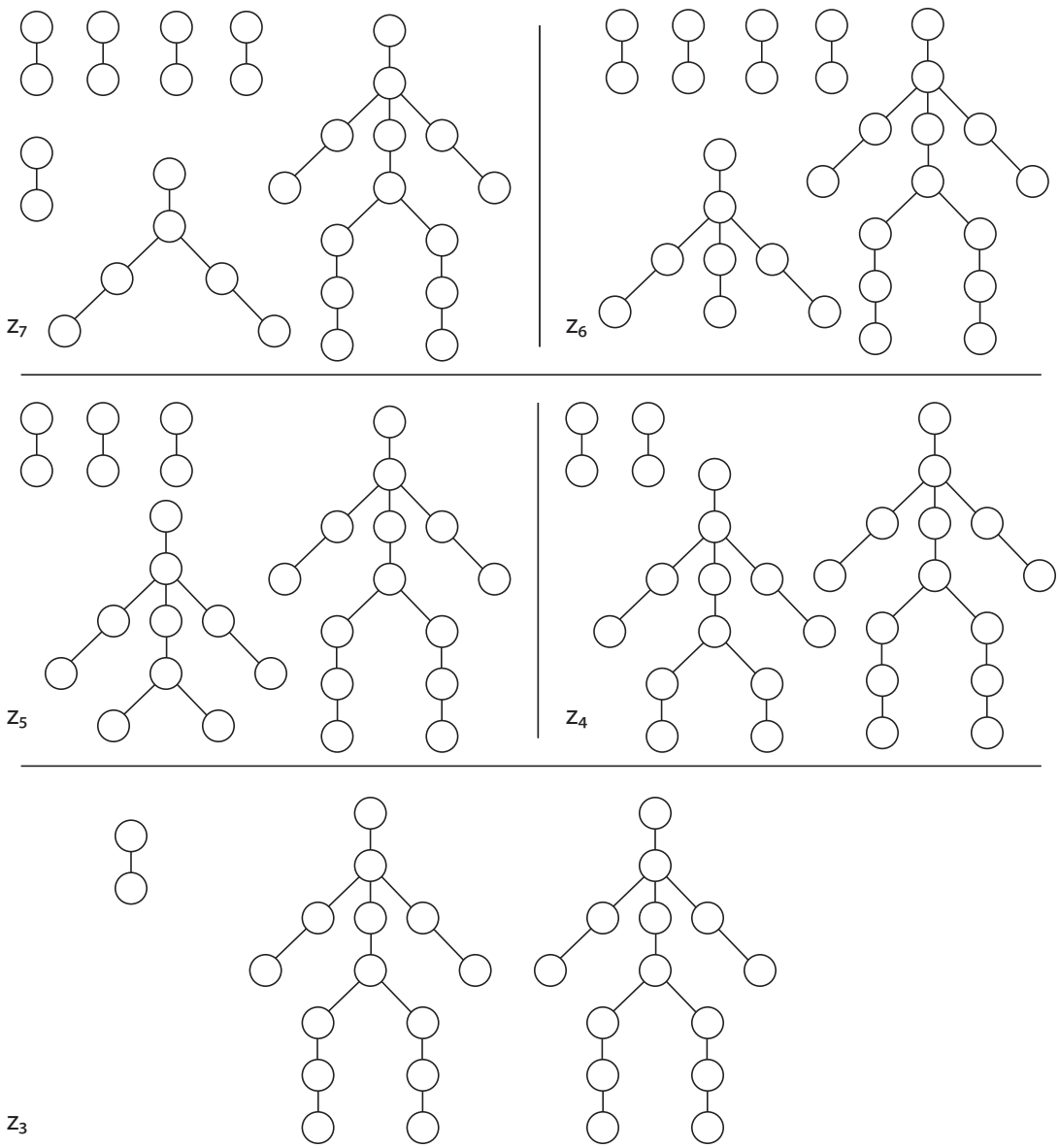

Figure 11: The states $z_m, m \in \{3, 4, ..., 7\}$ for the $\hat{G}$ introduced above, $N = 30$, $\hat{N} = 14$.

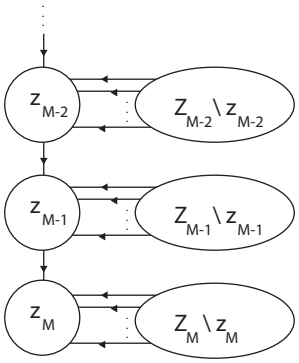

Figure 12: The structure of the tree rooted at $z_{\lceil N/\hat{N} \rceil}$; note that $M = \lceil N/\hat{N} \rceil$.

can construct the path to z_m for the two cases for y:

Case 1: There exists a maximal connected subgraph of y that is isomorphic to x_m. In this case, we take the smallest assembly with more than two nodes and shift a node to the largest incomplete assembly other than the one just mentioned that is isomorphic to x_m. We continue this process until we obtain z_m. Each step in the process involves one *break* and therefore an edge with $r = 1$ linking to a distinct absorbing state in Z_m.

Case 2: There is no maximal connected subgraph of y that is isomorphic to x_m. In this case, we take the smallest assembly with more than two nodes and shift a node to the largest maximal connected subgraph that is isomorphic to a subgraph of x_m. We proceed like this until we obtain a maximal connected subgraph that is isomorphic to x_m, and then continue as in case 1.

We can repeat this process until we have covered all of the states in Z_m, avoiding any redundancies so that we form precisely $|Z_m| - 1$ edges. Once we have applied this technique for all m we have obtained a rooted tree with each edge having resistance 1 so that $z_{\lceil N/\hat{N} \rceil}$ is stochastically stable.

A similar construction can be applied for all the other states in $Z_{\lceil N/\hat{N} \rceil}$. For an arbitrary state $z' \in Z_{\lceil N/\hat{N} \rceil}, z' \neq z_{\lceil N/\hat{N} \rceil}$ we construct the tree just as above for the states in the sets $Z_m, m > \lceil N/\hat{N} \rceil$ and $z_{\lceil N/\hat{N} \rceil}$. Then, we insert the edges between $z_{\lceil N/\hat{N} \rceil}$ and z' in the same way as above except that the directions are reversed. Then we apply the exact same

procedure as above for the remaining states in $Z_{\lceil N/\hat{N} \rceil}$, again avoiding any redundancies. These trees will also all have resistance 1 at every edge so that all of $Z_{\lceil N/\hat{N} \rceil}$ is stochastically stable.

For any state in $Z_m, m \neq \lceil N/\hat{N} \rceil$ the rooted trees all must include an edge that goes from a state with a smaller number of assemblies to a state with a larger number of assemblies. This can only be accomplished by two successive `break` actions corresponding to an edge with resistance 2. Since all other edges are at best resistance 1, all of these rooted trees have resistance equal to at least $|Z_0|$. We therefore conclude that the stochastically stable states are precisely $Z_{\lceil N/\hat{N} \rceil}$. ■

Corollary 5.1 ($N = m\hat{N}$): *If $N = m\hat{N}$ for some $m \in \mathbb{Z}^+$ then the only stochastically stable state is all maximal connected subgraphs of G isomorphic to $\hat{G}$.*

Proof: This follows directly from the previous claim. ■

When N is not an integer multiple of $\hat{N}$ then we do not expect all nodes to participate in complete assemblies. If we have N slightly larger than $\hat{N}$ then we cannot in general guarantee that the stochastically stable states will contain **any** complete assemblies. However, as N becomes large, the minimum number of completed assemblies in the stochastically stable states becomes large as well. In fact, there is a bound on the total number of nodes **not** participating in complete assemblies that applies for all N.

Theorem 5.1 (convergence to objective): *For any acyclic $\hat{G}$, all stochastically stables states of P^ϵ have no more than $(\hat{N}-1)^2$ distinct nodes not part of a connected subgraph isomorphic to $\hat{G}$.*

Proof: Let $N = (\hat{N}-1)^2$. The maximum number of incomplete assemblies is $\hat{N}-1$ assemblies with $\hat{N}-1$ nodes in each assembly. Each increase of N by one, must add one complete assembly and reduce the number of nodes not participating in complete assemblies by 13. This continues until we reach $N = \hat{N}(\hat{N}-1)$ and there are zero nodes not part of complete assemblies in the stochastically stable states. This process repeats for $\hat{N}(\hat{N}-1)+1$ through $\hat{N}^2$ and, it is easy to show by induction that $(\hat{N}-1)^2$ is always the

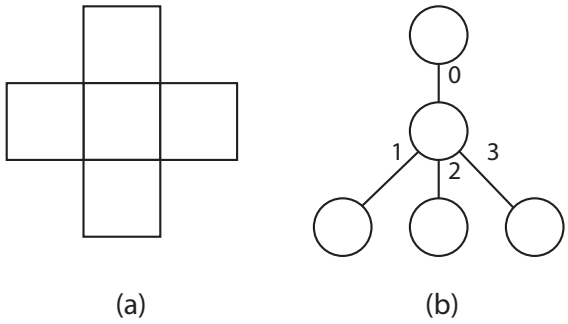

Figure 13: The assembly (a) for Example 5.1 and its associated graph (b).

maximum number of nodes not part of complete assemblies. ■

There are at most $\hat{N}-1$ incomplete assemblies (rejects). This result represents the thrust of this work. If N is much larger than $\hat{N}$, a reasonable expectation in many situations, then an assembly process that behaves like P^{ϵ} will self-assemble almost completely. The next section will present some examples and MATLAB simulations that serve to illustrate and verify the claims of this chapter.

5.5 Examples

Before we present MATLAB simulations for the $\hat{G}$ presented in the previous section, we will compute the exact stationary distribution for a simpler assembly task. Even for this simple case, the number of states is large and computation of the stationary distribution is somewhat laborious.

5.5.1 Example 5.1: Stationary distribution

To simplify the computation we will consider the assembly of Figure 13, a subgraph of the motivating assembly from the previous section. This graph can represent a subassembly of the assembly represented by the previously considered graph.

We will consider the initial condition of all $N = 5$ nodes having no connections. The probabilities for F will be uniform always. We have shown previously that all maximal subgraphs encountered will be isomorphic to a subgraph of $\hat{G}$. We will further simplify

Table 1: Reduced order representation for Example 5.1

Assembly sizes	State
$(1,1,1,1,1)$	s_1
$(2,1,1,1)$	s_2
$(2,2,1)$	s_3
$(3,1,1)$	s_4
$(3,2)$	s_5
$(4,1)$	s_6
(5)	s_7

the state space by grouping together states where the maximal subgraphs have the same number of nodes. The mapping is shown in Table 1. Figure 14 shows the three different assembly pairings that are represented by $(3,2)$, or s_5. The L-shaped assemblies in (a) and (c) are actually identical and can be attained by rotation. However, they are represented by two different graphs that are not isomorphic to each other. This coarsening of the state space will not impact our results significantly because there is no ambiguity associated with the desired state s_7. Since we intend to show convergence to s_7, the exact distributions for the other states is not important to us.

Figure 15 shows the transition probabilities for P^0, also representable in matrix form:

$$P^0 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{2}{5} & \frac{3}{10} & \frac{3}{10} & 0 & 0 & 0 \\ 0 & 0 & \frac{4}{5} & 0 & \frac{1}{5} & 0 & 0 \\ 0 & 0 & 0 & \frac{7}{10} & \frac{1}{10} & \frac{1}{5} & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{9}{10} & \frac{1}{10} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

It is easy to see that s_5, not s_7, is the slightly more probable outcome beginning from s_1. It can be verified that the state transition matrix for P^ϵ is as follows

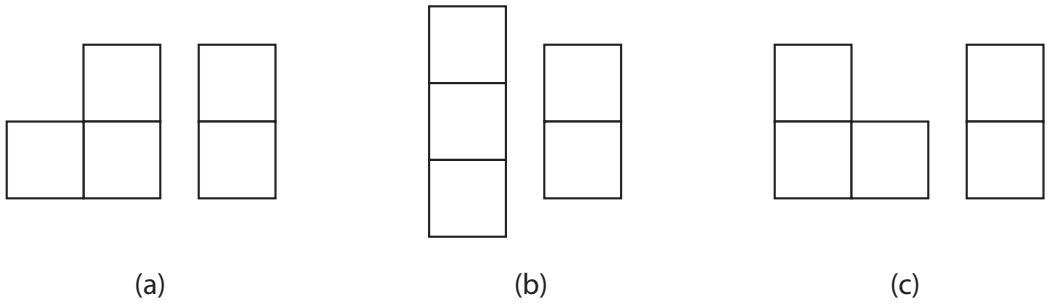

Figure 14: S_7 includes three different pairs of assemblies. (a) can be obtained from (c) by rotation, but their graphs are not isomorphic.

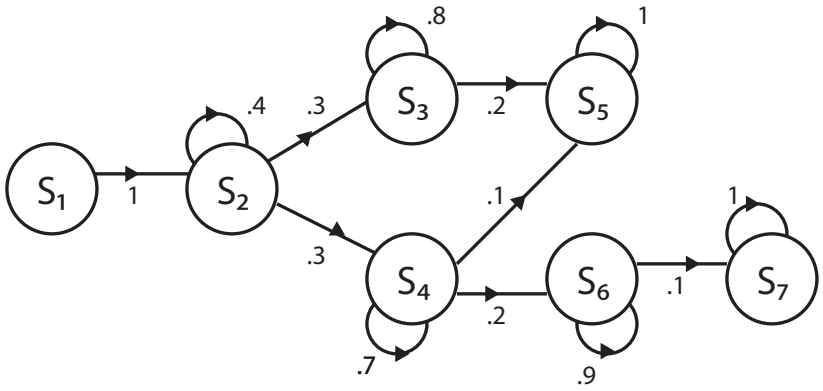

Figure 15: P^0 for Example 5.1. The recurrent classes are the absorbing states S_7 and S_5.

$$P^\epsilon = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{\epsilon}{10} & \frac{4}{10} - \frac{\epsilon}{10} & \frac{3}{10} & \frac{3}{10} & 0 & 0 & 0 \\ 0 & \frac{\epsilon}{5} & \frac{4}{5} - \frac{\epsilon}{5} & 0 & \frac{1}{5} & 0 & 0 \\ 0 & \frac{\epsilon}{10} & 0 & \frac{7}{10} - \frac{\epsilon}{10} & \frac{1}{10} & \frac{1}{5} & 0 \\ 0 & 0 & \frac{\epsilon}{10} & \frac{\epsilon}{10} & 1 - \frac{\epsilon}{5} & 0 & 0 \\ 0 & 0 & 0 & \frac{\epsilon}{5} & 0 & \frac{9}{10} - \frac{\epsilon}{5} & \frac{1}{10} \\ 0 & 0 & 0 & 0 & 0 & \frac{3\epsilon}{10} & 1 - \frac{3\epsilon}{10} \end{pmatrix}.$$

Table 2 gives the stationary distributions, π_ϵ for different values of ϵ. The stationary distribution satisfies $\pi_\epsilon = \pi_\epsilon P^\epsilon$. Notice that for $\epsilon = .05$, P^ϵ spends more than 99% of the time in the desired state, S_7.

Table 2: Stationary distributions for Example 5.1

ϵ	π_ϵ
.5	$(.0003, .0066, .0197, .0789, .1578, .3157, .4209)$
.3	$(0, .0005, .0026.0176, .0587, .1957, .7248)$
.1	$(0, 0, 0, .0003, .0029, .029, .9678)$
.05	$(0, 0, 0, 0, .0004, .0074, .9922)$

5.5.2 Example 5.2: $N = \hat{N}$

One difficulty associated with simulating the atom-level information self-assembly process is that the agent selection process F may commonly select agents whose only available action is `null`. In the simulations that follows, we choose F so that pairings that cannot result in a `form` or `break` are never selected. However we will choose F so that the relative probabilities for `break` and `form` are consistent with F uniform. The effect of this choice of F is that the absorbing states, where many pairings have only `null` available, are less often observed. The relative frequency of the different absorbing states is not substantially altered. Figure 16 shows the results of a simulation of $\hat{G}$ as in Figure 9 with $N = 14, \epsilon = .01$ beginning from the standard initial condition.

It takes a certain amount of time for the system to first reach the desired state so that the proportion of time spent there can increase. Afterwards, the system goes in and out of the desired state and begins to settle at a figure of about 50% of time being spent in the desired state. If ϵ is taken lower then the system will spend a higher proportion of total time in the desired state. However, the actual length of the time intervals in between will increase. While we may stay in the desired state for a very long period of time, it may take a very long time to get there. This is a limitation of the process presented and the convergence result achieved, but there are ways to improve performance other than just waiting longer.

We will employ a technique known as *cooling* to speed up the simulations. This involves beginning with a high value for ϵ (temperature) and gradually reducing it in order

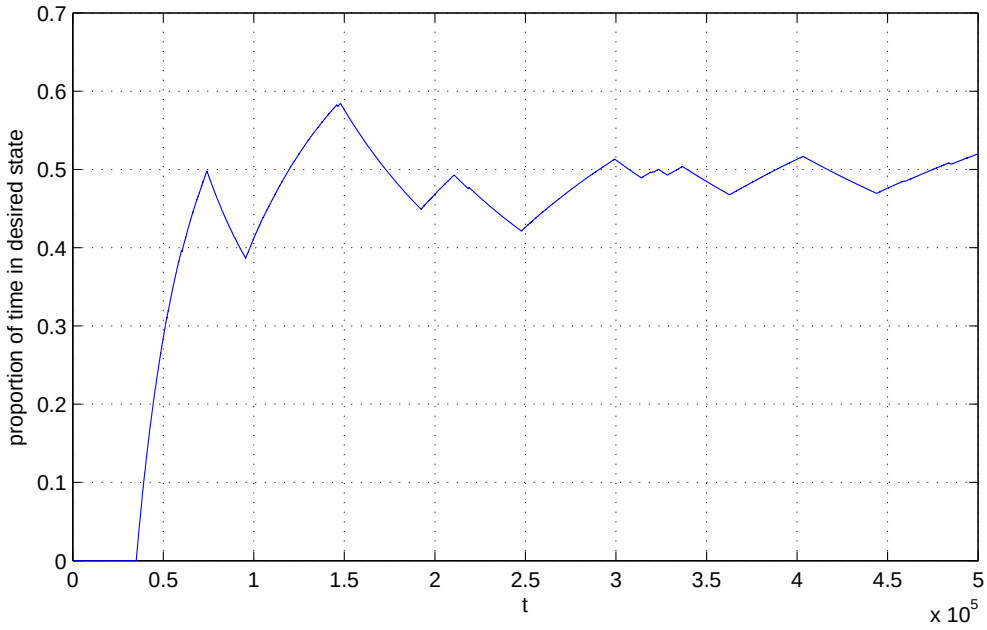

Figure 16: The proportion of total time such that $G \simeq \hat{G}$ for $N = 14, \epsilon = .01$ beginning from an initial condition where all atoms have no connections.

to settle into a stochastically stable state. One drawback of stochastic stability as a convergence result for engineered systems is that while ϵ can be made small to keep the system in the stochastically stable states a large proportion of the time, the absolute amount of time spent outside the stochastically stable states may be great. For example, a process may spend 90% of its time in stochastically stables states, spending an average of 90 minutes in the stochastically stable states, then 10 minutes outside them, and so on. We can increase this proportion to 99% by decreasing ϵ, but this may require spending an average of 99,000 minutes in the stochastically stable states, with recurring periods outside of the stochastically stable states averaging 1,000 minutes. For a task like self-assembly, such long waits may be unacceptable. Cooling can assuage these waiting times.

Figure 17 shows the dramatic performance improvement achieved using a simple linear evolution of ϵ. Over the time period $T = 5 \times 10^5$, ϵ is lowered to 0.001, a factor of 10 lower than the previous situation. In this case, we see that using higher values of ϵ early on

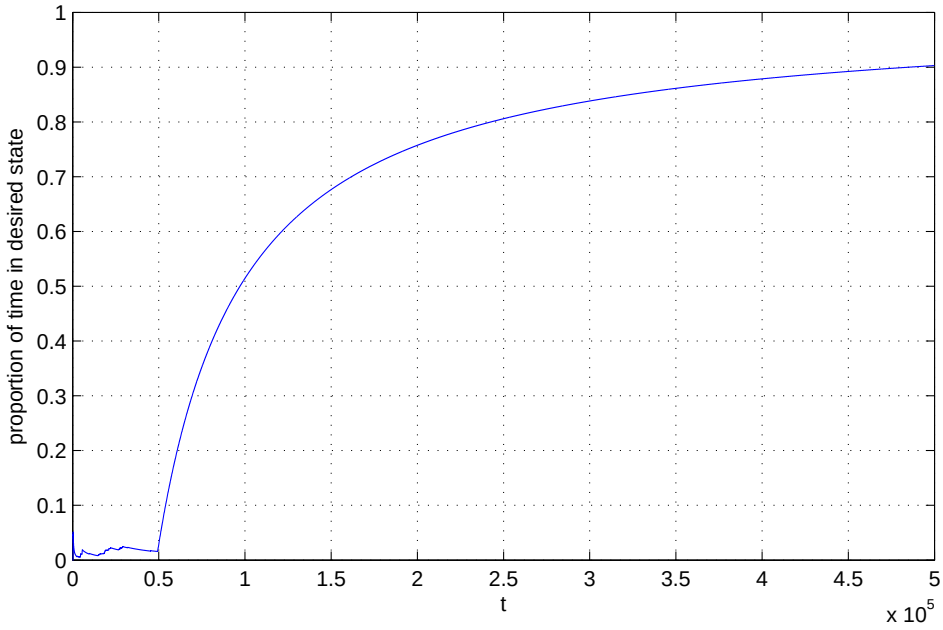

Figure 17: The proportion of total time such that $G \simeq \hat{G}$ for $N = 14$ using cooling: $\epsilon = 0.5 - 0.499(t/5 \times 10^5)$.

allows us to reach the desired state quicker. Then, using lower values of ϵ once there is a high probability of having reached the desired state lowers the probability that the system will leave it. We see a dramatic improvement in the likelihood of observing the desired state when the system undergoes cooling.

5.5.3 Cooling performance

Discussion of optimal cooling techniques is beyond the scope of this work. In general, the right technique will depend on the demands of the specific application. In the previous subsection we showed how a cooling technique could be used to improve the likelihood of observing the desired state over a fixed length of time. In this subsection we will instead attempt to cool a system so that at one specific time $t = T$, the system is in the desired state. Continuing with $\hat{G}$ as before, we will vary the time T and simulate the system 50 times for each value. The parameter ϵ will obey

$$\epsilon = 0.001 + 0.5e^{(-\frac{10t}{T})}$$

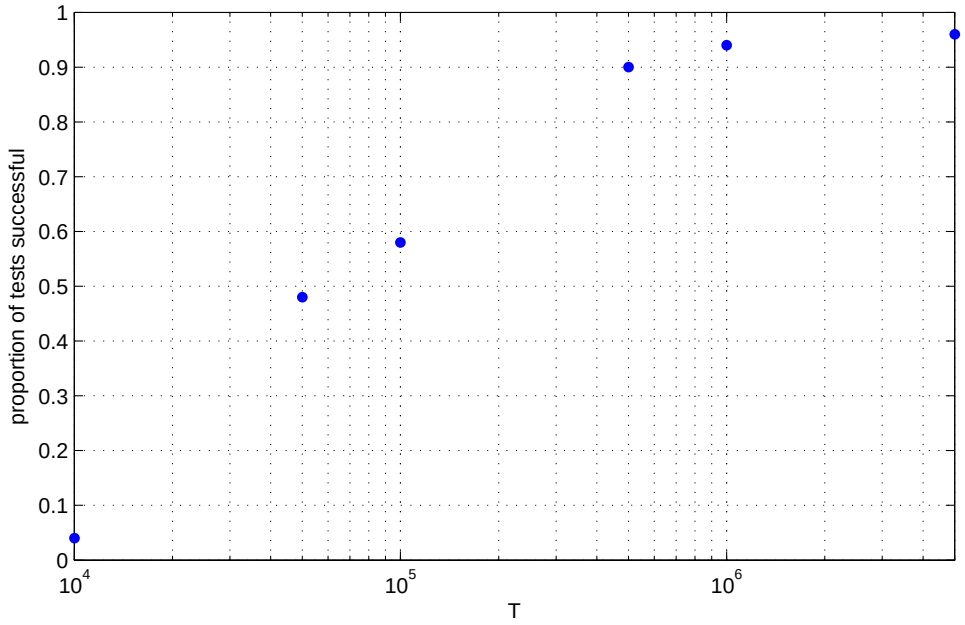

Figure 18: The proportion of the 50 tests for each T such that $G \simeq \hat{G}$ at time T for $N = 14$ using cooling: $\epsilon = 0.001 + 0.5e^{(-\frac{10t}{T})}$.

for each T. This way we maintain the same final ϵ of approximately 0.001. We compare the likelihood of the system being in the desired state at $t = T$ as we cool the system for longer and longer lengths of time. Figure 18 shows the results of this simulation. Cooling the system more slowly leads to a higher probability of observing the desired state at the sample time. Since $\epsilon > 0$ for all t, the probability of observing the desired state at T will not go to 1 as T is increased. Nevertheless, we see that 48 out of 50 trials were successful for $T = 5 \times 10^6$. Table 3 presents the results of Figure 18.

The minimum assembly count property guarantees similar results when $N = m\hat{N}$ for some $m \in \mathbb{Z}^+$. However, the same cannot be said for cases when N is not an integer multiple of $\hat{N}$. The next example illustrates what sort of performance can be expected in this situation.

Table 3: Data for Figure 18

T	# successful (out of 50)	rate
10^4	2	0.04
5×10^4	24	0.48
10^5	29	0.58
5×10^5	45	0.90
10^6	47	0.94
5×10^6	48	0.96

5.5.4 Example 5.3: N not an integer multiple of $\hat{N}$

This example will highlight the shortcoming of the presented process when N is small, and how it is overcome when N is large. The simulations in this example and all others in this thesis utilize an approximation technique in order to speed up their running time. Whenever the system is in an absorbing state only the `break` action is available. However, if ϵ is small, the action will usually be taken many times before it results in the actual severance of a bond. For this reason, whenever the system is in one of these states, one of the breakable bonds is automatically broken at random with the expected waiting time, $1/\epsilon$ in this case, added to the total time.

5.5.4.1 N small

When $N \neq m\hat{N}$ for all $m \in \mathbb{Z}^+$, many different G will satisfy the minimum assembly count property and not all of these will contain maximal connected subgraphs isomorphic to $\hat{G}$ in general. For example, consider the previous example with $N = 16$. In this case, all states that are two disjoint maximal connected subgraphs are stochastically stable. Many of these states do not contain a maximal connected subgraph isomorphic to $\hat{G}$. The methods presented in this work do not perform well when N is neither large or an integer multiple of $\hat{N}$. We will here treat the case when N is large and show that self-assembly can be achieved even when $N \neq m\hat{N}$. In the next chapter we will revisit the case of N small, but not an integer multiple of $\hat{N}$ and suggest some approaches to improve performance.

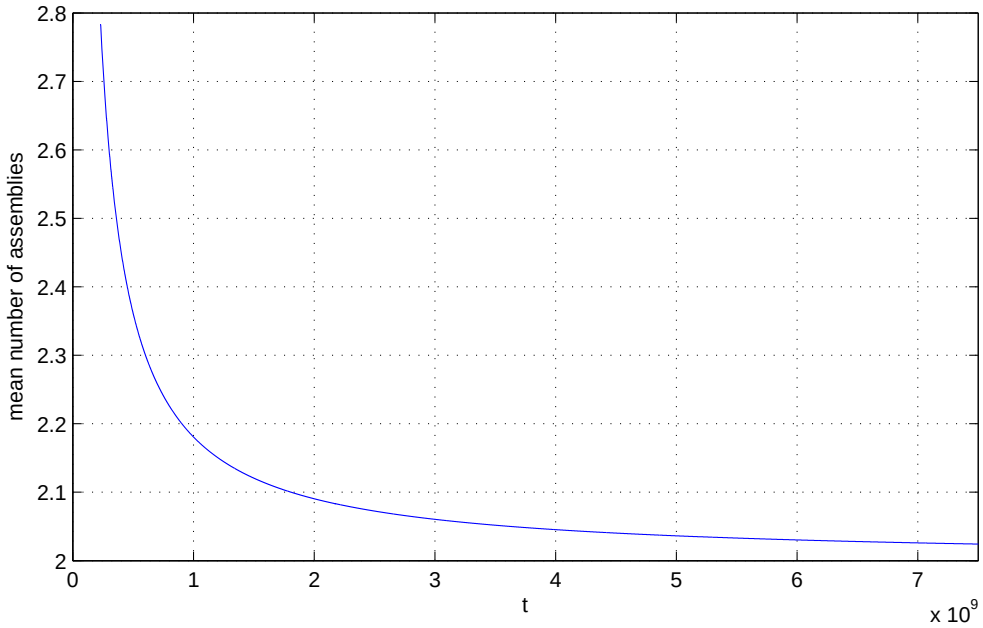

Figure 19: With $N = 28, \epsilon = 0.1e^{(-\frac{t}{10^5})} + 10^{-5}$ the system achieves a mean number of assemblies (for all t) of less than 2.03 after 7.5×10^9 iterations.

We again consider the same $\hat{G}$ as in the previous example. For the case of $N = 28$ we expect the stochastically stable states to all feature two completed assemblies. This is a result of the minimum assembly count property. Whenever there are two assemblies, all of the pieces must participate in the two complete assemblies. If there are more assemblies, there may be at most one complete assembly. For this reason, we examine the average over all t of the number of maximal connected subgraphs for each time t. We expect this average to approach a number very close to 2 when ϵ is small. This is equivalent to there being a very high probability of observing exactly two (the maximum) completed assemblies. Figure 19 illustrates the result for $\epsilon = 0.1e^{(-\frac{t}{10^5})} + 10^{-5}$. Although the system actually continues to go in and out of the desired state for all t, the average appears to decrease monotonically because *most* time is spent in the desired state, and the difference between the average number of assemblies and two is largely the result of the large number of assemblies while t is small.

If we increase N by one so that $N = 29$, the situation changes dramatically. Now, the minimum number of assemblies is three. Most of the stochastically stable states do not include *any* completed assemblies. No self-assembly guarantee can be made at all, regardless of our selection of ϵ or the time elapsed. For $N = 41$, the minimum number of assemblies is still three. In this case though, there can be only one assembly that is not complete, and it can only lack one piece. Therefore, when $N = 41$, the stochastically stable states all feature two complete assemblies- the maximum possible. Nevertheless, the sensitivity of system performance to variation in N when N is small is undesirable. This sensitivity diminishes though as N becomes large relative to $\hat{N}$.

5.5.4.2 N large

For N large, Theorem 5.1 implies that the minimum number of assemblies in all stochastically stables states increases by one for every increase of N by $\hat{N}$. This implies one more completed assembly in all stochastically stable states. We simulate this phenomenon for the $\hat{G}$ of example 5.1. In this case $\hat{N} = 5$ so that $(\hat{N} - 1)^2 = 16$. Self-assembly performance is at its worst when $N = m\hat{N} + 1$ for some $m \in \mathbb{Z}^+$. This is because the minimum number of assemblies is $m + 1$, so that $\hat{N} - 1$ more pieces are needed to arrive at all assemblies completed ($N = (m + 1)\hat{N}$). It is possible that as many as $\hat{N} - 1$ assemblies are incomplete, this being achieved when each of those assemblies has $\hat{N} - 1$ parts. This is why there exist stochastically stable states for $N \leq (\hat{N} - 1)^2$ having no completed assemblies. We consider the case of $N = 20\hat{N} + 1 = 101$. We expect all stochastically stable states of the system to have the minimum number of assemblies, 21. At most 4 assemblies are incomplete so that whenever the system has 21 assemblies, at least 17 are complete. Figure 20 shows the results of this simulation for $\epsilon = 0.1e^{(-\frac{t}{10^6})} + 10^{-6}$.

Figure 20 illustrates the self-assembly process's typical 'driving down' of the number of assemblies over time. The mean number of assemblies first seems to settle in at 22 and then later abruptly begins to approach 21, the value exhibited by stochastically stable states.

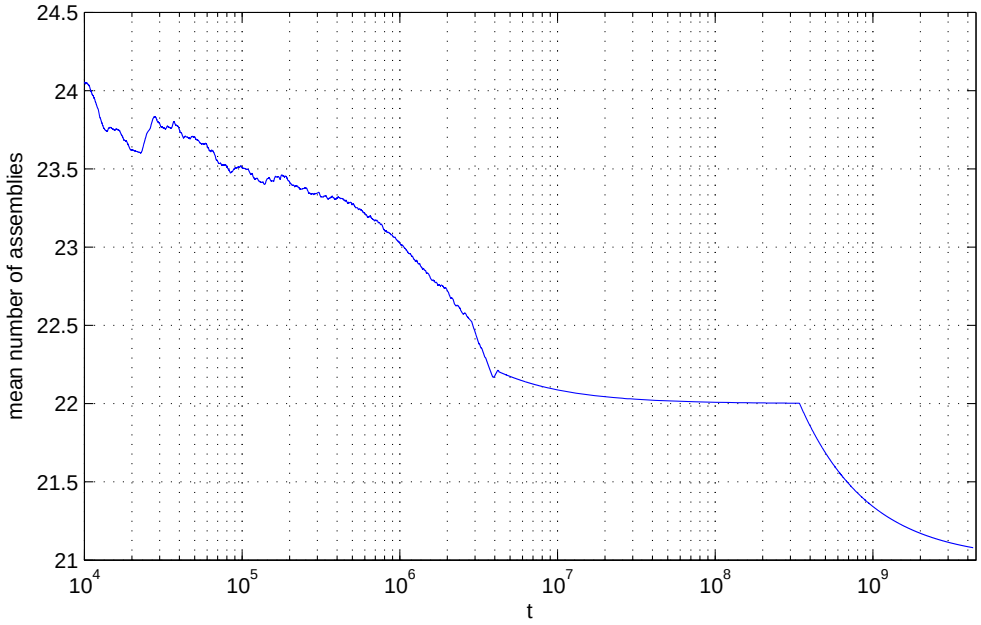

Figure 20: With $N = 101, \epsilon = 0.1e^{(-\frac{t}{10^6})} + 10^{-6}$ the system achieves a mean number of assemblies (for all t) of less than 21.08 after 4.3×10^9 iterations.

Since at this stage, ϵ is very close to 10^{-6}, `break` actions are quite improbable. The system spends nearly all of its time in the absorbing states of P^0- all nodes having at least one edge. It will take, on average, about $1/\epsilon = 10^6$ iterations before the `break` action results in the elimination of an edge and the production of a free node. Then, the probability of that free node rejoining one of the other assemblies is much greater than another node breaking off, so that is what we expect to happen almost always. If we are oscillating in this manner about a non-minimum number of assemblies, this process continues until eventually the final two nodes of an assembly are broken apart. There is then a reasonable probability that these two nodes will each add onto existing assemblies, resulting in a state with one less assembly.

Figure 21 provides the raw number of assemblies at each time iteration for the same simulation. While this is too much data to make quantitative observations, it does facilitate a qualitative observation. During the earlier time iterations while ϵ is large, we observe that

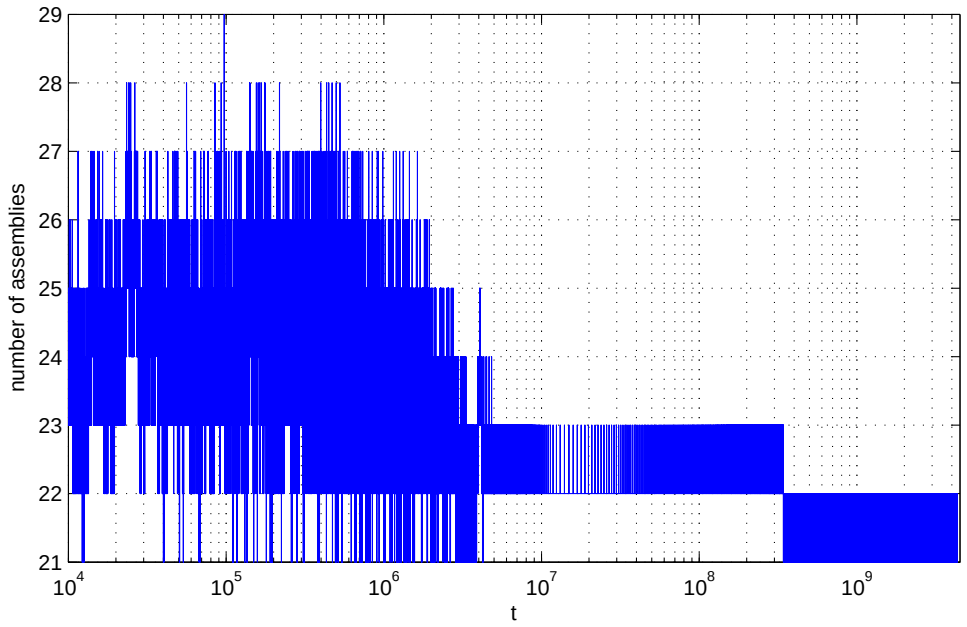

Figure 21: The raw number of assemblies will settle into oscillation between 22 and 23, and then 22 and 21.

the system transits frequently between different number of assemblies. By $t = 10^7$ the system begins to transition between 23 and 22 assemblies exclusively. The process described above is occurring here. The system is spending almost all its time with 22 assemblies, breaking off a piece and then reforming it again and again. Eventually, a favorable series of outcomes leads to the breakup of a two-node assembly and assimilation of the two free nodes by existing assemblies. This results in 21 assemblies. The process of breaking up and rejoining continues, with the system spending nearly all of its time with 21 assemblies. It is very unlikely that the system will increase the number of assemblies because this would require two successive `break` actions without a `form` action in between.

If we were to continue increasing N, the number of completed assemblies in the stochastically stable states would increase and the number of incomplete assemblies would always be 4 or fewer. By making ϵ small, we can achieve a probability of observing the minimum number of assemblies that is as close to one as we please. The difficulty with N large and ϵ small however, is that the number of iterations required to achieve satisfactory results can

be very long.

It should be noted that ‘time’ in our models is a relative concept. If the ‘parts’ are in a fixed volume (e.g. particles in a test-tube), then we should expect the rate that interactions occur to increase with N. Our procedure considers one interaction per iteration though, making large N cases appear to evolve very slowly. Another issue is that the procedure presented in this work is formulated so as to make the presentation of analytical results as clear as possible. In particular, we use resistances of 1 for every breakup. In the next chapter we will explore selecting different resistances for different edges empirically to improve performance.

CHAPTER VI

EXTENSIONS

In this chapter we will suggest modifications to the atom-level process that can improve performance. We also study and improve upon the process's sensitivity to some simple disturbances. First, we discuss how the system can be modified so that errors can be corrected.

6.1 Error correction

All the states of the process presented in the last chapter are composed of maximal connected subgraphs that are each isomorphic to a subgraph of $\hat{G}$. That is, the assemblies are always sub-assemblies of the desired assembly. This is an outcome of the dynamics, and the assumption that the system is initialized with all parts disconnected. The same holds if the system is initialized in any state within this invariant set. The system can be modified so that this result holds from *any* initial condition, even if there exists an assembly that is not a subassembly of the desired assembly.

In order to achieve this, we will modify P^0. We introduce an action, `correct`, which allows agents to unilaterally sever bonds they know to be mistaken. The action `correct` is given preference over all other actions in P^0. That is, if the criteria for the action are met it is taken, even if another action could be taken. We describe the criteria for the availability of the `correct` action as well as the result of the action for agent i. If the action is not available to i then it should be considered for j as well. We will make the following assumption about $\hat{G}$, which sacrifices no generality.

Assumption 6.1 ($\hat{N}$ increasing): *In $\hat{G}$, any path of distinct, connected vertices $\{v_k\}_{k=1}^{K} \in \hat{N}$ satisfies $v_k > v_{k-1},\ k = 2, 3, ..., K$.*

Figure 9(b) provides an example of a graph satisfying the assumption. An outcome of this structure for $\hat{G}$ is that a node i with $S_i(t) \neq 1$ and $|n_{G(t)}(i)| \geq 1$ must share an edge

with a node j having $S_j(t) < S_i(t)$. We now present the `correct` action.

- `correct`: (i) Let $L_1 \subset \mathcal{N} - \{i, j\}$ be the largest set such that each $l \in L_1$ has $(i, l) \in E(t)$, but either $(S_i(t), S_l(t)) \notin \hat{E}$ or $W(i, j) \neq \hat{W}(S_i(t), S_l(t))$. (ii) Additionally, if there is no $k \in n_{G(t)}(i)$ with both $S_k(t) < S_i(t)$ and $W(k, i) = \hat{W}(S_k(t), S_i(t))$; then let $L_2 = \{l : (i, l) \in E(t)\}$, otherwise L_2 is empty. Set

$$E(t+1) = E(t) - L_1 \cup L_2$$

  Additionally, if L_2 is nonempty, set

$$\begin{cases} S_i(t+1) = 0 \\ S_k(t+1) = S_k(t) \quad k \neq i \end{cases}$$

- If $L_1 \cup L_2$ is empty, proceed with P^0 as before.

Remember that we check for the `correct` action for i and then j. The correction (i) unilaterally eliminates all incorrect connections. The second correction (ii) determines if a node lacks connection with the proper parent node, as specified by $\hat{G}$. We correct both (i) and (ii) if possible. An important feature of the `correct` action is that it is enacted unilaterally- a major advantage in the presence of malfunctioning or malicious agents. The modified P^0 has a non-zero probability path from any state to the invariant set of states having all maximal connected subgraphs isomorphic to a subgraph of $\hat{G}$. Since the absorbing states of P^0 are the same and we have not changed the resistances in P^ϵ, this system achieves all of the same results as the unmodified process, but from any arbitrary initial condition.

The `correct` action provides a simple remedy for certain types of errors in the assembly process. We can model errors as perturbations to P^0. Consider a (not necessarily regular) perturbation of P^0, $\tilde{P}^0$. Suppose that there is a (fixed) probability $\delta > 0$ that following a `form` action, the new atom's state is updated incorrectly. One of the two atoms

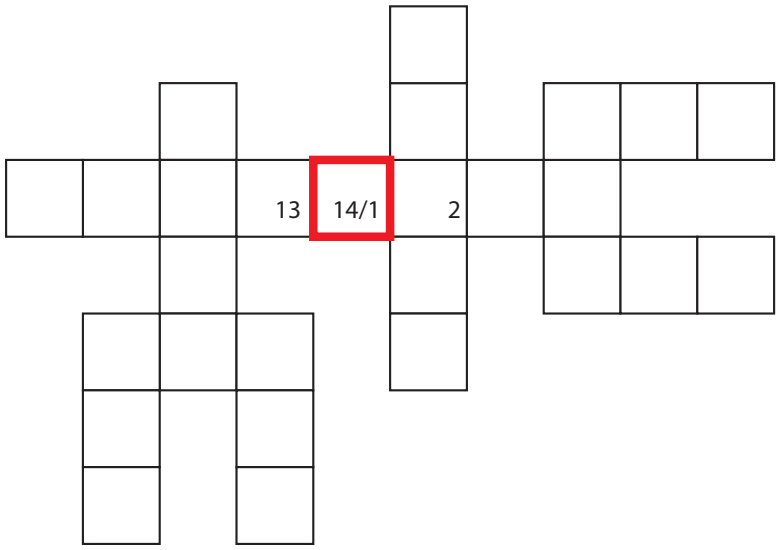

Figure 22: The malicious agent (outlined in red) is able to circumvent the *correct* action by falsifying its internal state to its neighbors.

will eventually perform the `correct` action and evacuate that bond. If the evacuated node has since formed additional bonds, it will eventually sever those as it realizes its error. The atoms connected to it will sever their bonds in turn when they see they have lost the necessary parent node until the entire erroneous appendage has been eliminated. $\tilde{P}^0$ has the same absorbing states as P^0 and can be perturbed as $\tilde{P}^\epsilon$ producing the same results as above. Stochastic stability is determined entirely by the recurrent classes and the resistances between them. This implies that any sort of error that does not change these relationships will not change the stochastically stable states.

The `correct` action does not however render the self-assembly process immunes to all types of error or manipulation. As an example, consider the presence of a malicious agent that is able to represent its internal state differently to different agents. It forms a bond and appears to have the proper resulting state, but then advertises a different state to other agents afterwards. Since the exploited agents do not know about the malicious agent's other bonds, they do not know they are participating in a perpetually flawed structure. Figure 22 gives an example using the familiar $\hat{G}$ from the last chapter. The malicious agent (outlined

in red) attaches to a node with the state 13 and adopts the appropriate state of 14. It later represents itself as a state 0, and then forms with another 0, updating its state to 1 for the purposes of that connection.

The figure illustrates that if a malicious agent can represent its internal state in multiple ways, the neighboring agents will have no basis to suspect that their reliance on local information is being exploited. Since the `correct` action will not rectify this situation, the structure in Figure 22 is part of an absorbing state. Since the absorbing states have been altered by the behavior of the malicious agent, the performance guarantees of the previous chapter no longer apply. In this case, the presence of a single malicious agent reduces the expected number of nodes participating in completed assemblies by as much as $2\hat{N} - 1$. This will be insignificant if the number of malicious agents is much less than N.

The form of disturbance that is most realistic will depend on the specific application. Here, we intend only to suggest that the process presented is not unreasonably sensitive to the agents' ability to follow the rules. The impact of agents acting correctly only a certain percentage of the time will usually be to require more iterations and a smaller value of ϵ for the same level of self-assembly performance as the analogous process without errors. Even when the disturbance is specifically engineered to disrupt the process, as with the malicious agent, the true distributed nature of the process prevents circumvention of the entire population by a small number of agents acting in a deleterious way.

6.2 Heterogeneous resistances

In the presentation of the last chapter, the `break` action, once undertaken, resulted in the elimination of an edge with probability ϵ. This strategy leads to convergence results that are straightforward to demonstrate analytically, but are not optimal with respect to convergence rates or yield. We have thus far looked at simulations that highlight the large number of iterations required to observe the proposed convergence, particularly when N is large. In the discussion that follows, we propose simple, empirical state-dependent probabilities

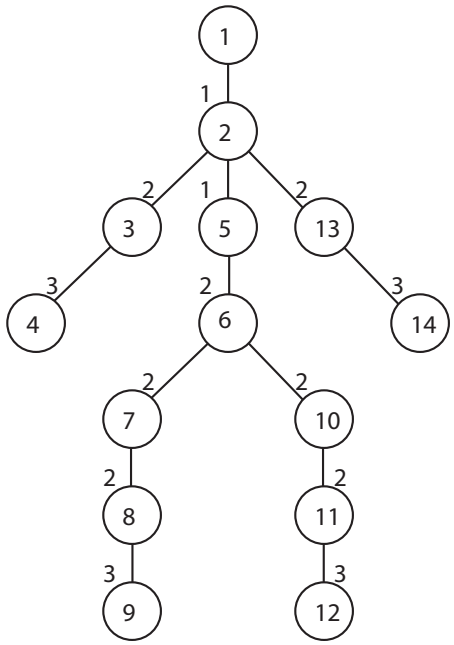

Figure 23: Assigning more resistance to the breakup of more advanced bonds improves performance empirically.

associated with the `break` action's elimination of edges for the motivating example of the previous chapter. We have not verified these results analytically, but consider simulation results to be suggestive that the same convergence results apply. The existence of a deterministic procedure for selecting the breakup probabilities for general $\hat{G}$ is also an open question.

We assign breakup probabilities based on the two internal states of the nodes undertaking the `break` action, so that the restriction of atom-level information is maintained. Figure 23 indicates the breakup probabilities (in powers of ϵ) associated with each pair of internal states. The idea is to deter breakup of the more complete assemblies, so that the smaller assemblies can be preferred and the number of assemblies can decrease faster. To develop a rigorous proof of convergence for this version of the process, we would have to consider resistance trees with edges having resistances different from 1 and 2. Alternatively, the process could be formulated as a potential game with the potential function being the total number of edges. While such a proof is not yet known to us, we have an

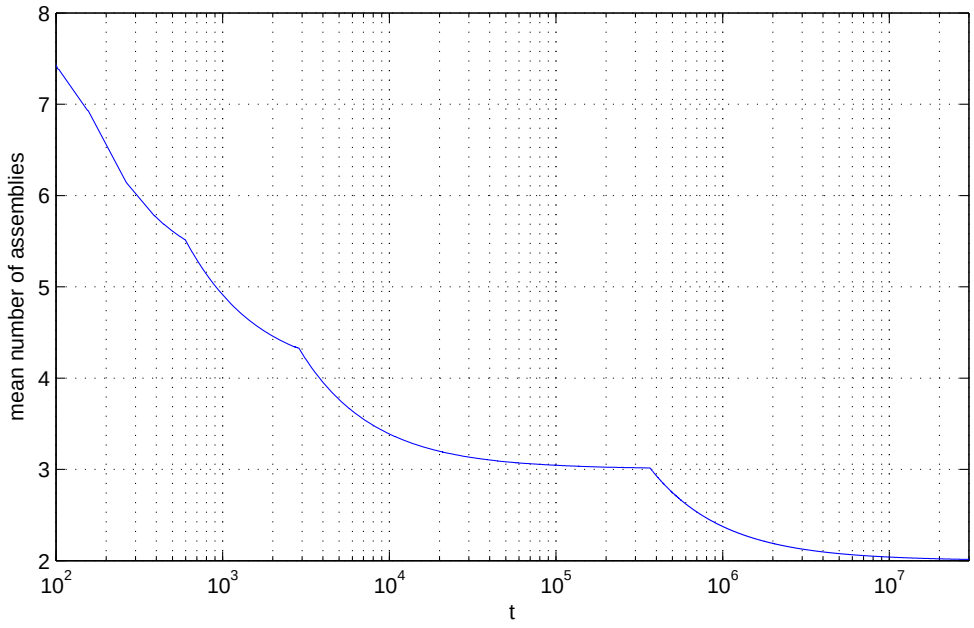

Figure 24: The convergence rate appears to be much faster with the empirical breakup probabilities.

indication of the efficacy of using these resistances from MATLAB simulation.

We first revisit the situation in the first part of example 5.3. There, the average number of assemblies got close to the minimum number, 2, but took 7.5×10^9 iterations to do so. When we instead use the heterogeneous breakup probabilities from Figure 23, the system responds much faster. Figure 24 illustrates this result using $\epsilon = \max\{0.1(1-t^{.25}/100), 0\} + 10^{-4}$. We are able to use larger values of ϵ since we will generally be raising it to one of the integer powers. Here, the system reaches an average number of assemblies of less than 2.03 in just over 10^7 iterations.

To strengthen the case for heterogeneous breakup probabilities, we provide the results of a simulation for $N = 1000$, using the usual $\hat{G}$. Since the methods presented in this work have special relevance to systems with large numbers of agents, we consider the positive results of this simulation important, despite the lack of theoretical justification. Figure 25 shows the results of the simulation.

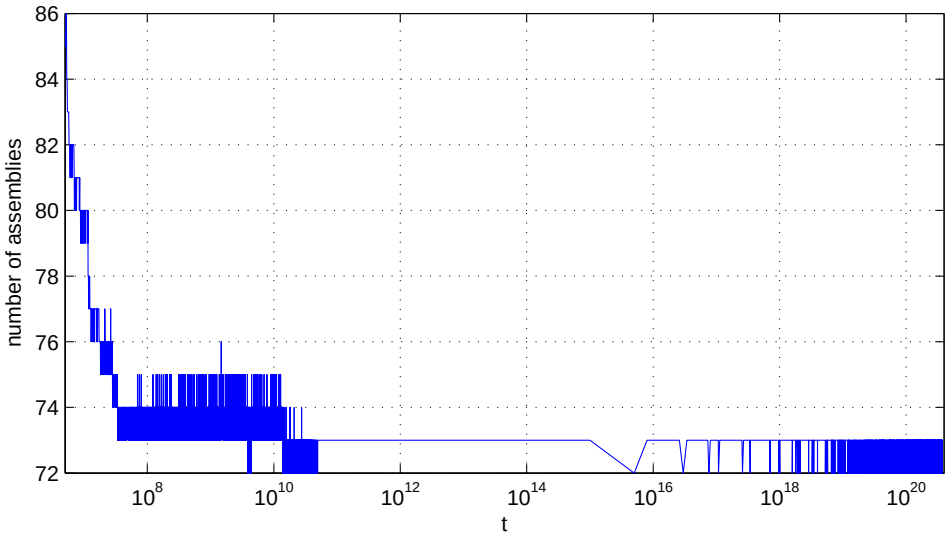

Figure 25: $N = 1000, \epsilon = \max\{0.1(1 - t^{.25}/500), 0\} + 10^{-5}$. The system converges to oscillation about the minimum number of assemblies, 72.

Another apparent advantage of heterogeneous resistances is related to the yield of desirable assemblies. With homogeneous resistances the yield was related to the stochastically stable states having the minimum number of assemblies. This guaranteed no more than $\hat{N} - 1$ rejects, but we could not expect to see desirable assemblies when the minimum number of assemblies was less than $\hat{N} - 1$ (and not an integer). For example, using the same $\hat{G}$ as above with $\hat{N} = 14$, if we use $N = 15$ parts, we would expect just two assemblies in the stochastically stable stats, but not necessarily any complete assemblies. This is not the case with heterogeneous resistances. While we stress that these results have not been verified analytically, preliminary simulation results suggest that carefully picked heterogeneous resistances can achieve guarantees of fewer than $\hat{N} - 1$ rejects.

We consider $N = 15$ using the same $\hat{G}$ with heterogeneous resistances as above. A simulation using a $\epsilon = \max\{0.1(1 - t^{.25}/50), 0\} + 10^{-4}$ converges to an average number of complete assemblies greater than 0.92, the maximum number being 1. This phenomenon is not limited to this case. We have observed convergence with fewer than $\hat{N} - 1$ rejects for many different values of N, although the system does not converge to the maximum

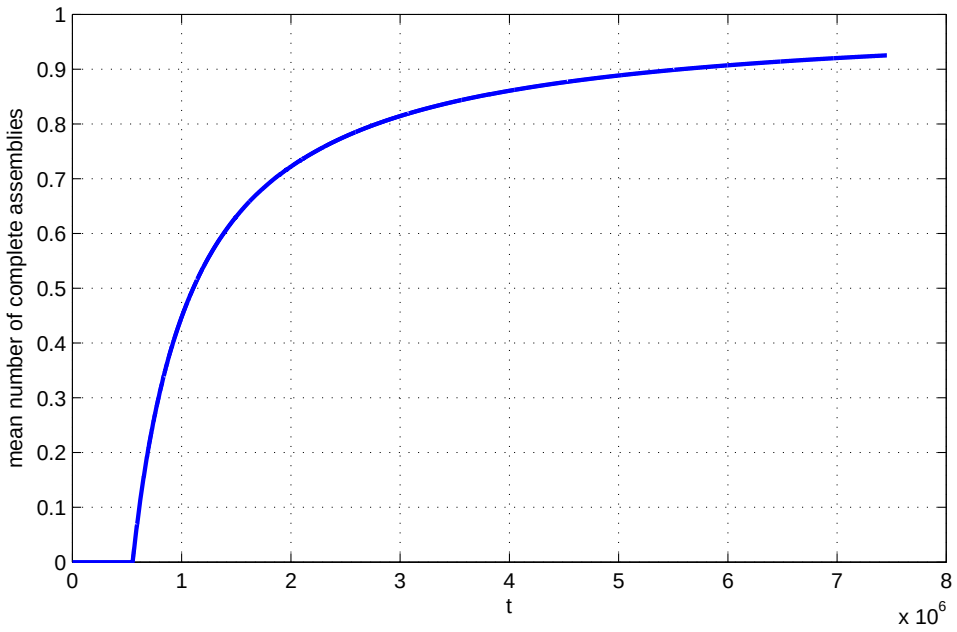

Figure 26: With heterogeneous resistances, the usual $\hat{G}$, and $N = 15$ the system still converges to close to one completed assembly, the maximum number.

number of assemblies (as it does here) for all such values. Whether or not resistances can be selected that can have the maximum number of desirable assemblies in all stochastically stable states for all $N \geq \hat{N}$ is an open question.

6.3 Feedback control

This thesis answers questions about self-assembly when the requirement of local rules is interpreted in a quite rigid way. It is clear from the assembly-level information case that relaxation of information constraints on the policies is extremely material to self-assembly performance. One can imagine a continuum of self-assembly policies corresponding to problems falling between our two cases. One particularly interesting aspect to consider is the parameter ϵ in the atom-level case. We have heretofore looked at this parameter as exogenous to the process. Either it is a fixed number, or monotonically decreases as time evolves. Suppose that ϵ can be controlled and made to depend on the state of the system. This approach shares similarities with [21]. This concept is illustrated in Figure 27.

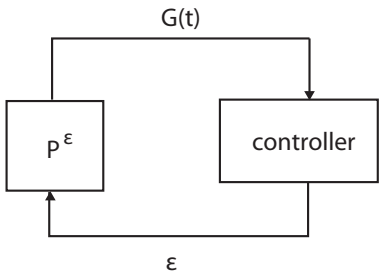

Figure 27: A feedback configuration where $G(t)$ is sampled and the parameter ϵ is adjusted appropriately.

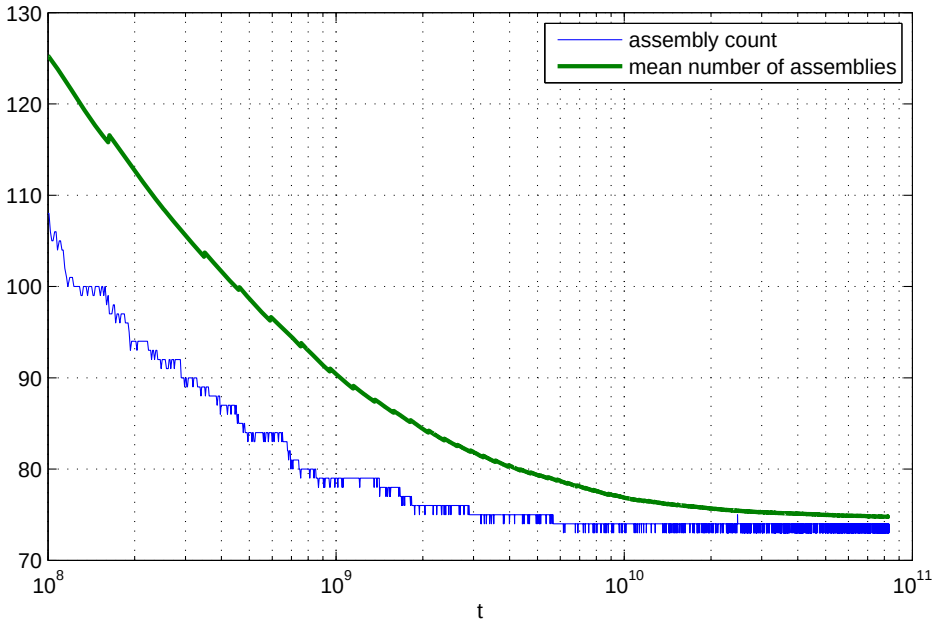

Figure 28: Using feedback with $N = 1000$, the system converges to the minimum number of assemblies, 72, quickly and in nearly monotone fashion.

The advantage of controlling ϵ based on $G(t)$ is that we can have ϵ be decreasing in the number of completed assemblies. When ϵ is varied as a function of time this phenomenon is sought as a result of the dynamics. Controlling ϵ in this way intuitively gives faster convergence. If we can detect when the maximum number of completed assemblies is present (without delay) then we can even set ϵ to zero at that moment and make invariant the state where the maximum number of completed assemblies is present, giving almost sure convergence. Recall that without feedback we were limited to stochastic stability of states with minimum assembly count.

Figure 28 shows the result of a simulation using the same $\hat{G}$ as above with the heterogeneous resistances. Here $\epsilon = (\frac{n(t)-72}{10^8})^{1/4}$. Once the system reaches 72 assemblies, it will set $\epsilon = 0$ and remain in that state forever. For this reason, we end the simulation at that moment.

The addition of feedback gives "closer" to monotonic increase of the number of desirable assemblies with respect to time. Recall that such monotonicity, a long with invariance of the desired state, were key features of the assembly-level information case. This 'in-between' case of atom-level information with ϵ fed back in suggests yet again a strong relationship between information and performance and the existence of a continuum of self-assembly systems allowing for a fine trade-off between these two factors appropriate to specific applications.

CHAPTER VII

CONCLUSIONS

Self-assembly with local rules has important implications for many fields. Still, results exploring foundational questions about self-assembly in a theoretical context have been few and far between. In this thesis, we have presented the first ever self-assembly results with restrictions on intra-assembly communication. We also introduced the notion of a probabilistic performance guarantee to the area. That is, our atom-level process's long-run behavior is described by stochastic stability. Since simplicity of the self-assembly policy is a critical factor, the development of processes like ours with memory and computation requirements that are linear in the size of the desired assembly is a relevant direction for research.

The assembly-level case highlights the role that information plays in self-assembly performance. While self-assembly in that context is not new, we presented it in a streamlined manner that nuanced the atom-level information case. It remains an open question whether there is any way to achieve the stronger form of convergence, almost sure, in the atom-level case.

The most important future step for this research is to contextualize. In what applications can the ideas here make a tangible impact? What are the specific challenges for implementation? Self-assembly is ultimately concerned with building things and theoretical results must derive their importance through affect in that context. This is an ongoing question not just for our work, but for everyone advancing the abstract theory of self-assembly.

We have suggested several directions already in which this research can be continued. From a theoretical standpoint we are very interested in methods of proving the stochastic stability of self-assembly processes with heterogeneous breakup probabilities. We have

considered extensions to the work where information about the system can be fed back in, or rules can be augmented to handle errors. Ultimately, these sort of innovations may or may not be of real import depending on the context, if any, in which the general ideas here most appropriately apply.

REFERENCES

[1] J. von Neumann, "The general and logical theory of automata", *Cerebral Mechanisms in Behavior The Hixon Symposium*, John Wiley & Sons, New York, 1951, pp. 1-31.

[2] J. von Neumann, "Re-evaluation of the Problems of Complicated Automata Problems of Hierarchy and Evolution, Fifth Lecture, University of Illinois, December 1949.

[3] J.G. Kemeny, "Man Viewed as a Machine". *Scientific American* 192, Apr. 1955: pp. 5867.

[4] K. Kotay and D. Rus, "Generic Distributed Assembly and Repair Algorithms for Self-Reconfiguring Robots", *IEEE Intl. Conf. on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[5] K. Fujibayashi, R. Hariadi, S. Park, E. Winfree, and S. Murata, "Toward Reliable Algorithmic Self-Assembly of DNA Tiles: A Fixed-Width Cellular Automaton Pattern", *Nano Letters*, 2008 8 (7), pp. 1791-1797.

[6] M. Gardner, "Mathematical Games: The fantastic combinations of John Conway's new solitaire game "life"", *Scientific American* 223, Oct. 1970, pp. 120-123.

[7] NASA Conference Publication 2255 (1982), based on the Advanced Automation for Space Missions NASA/ASEE summer study Held at the University of Santa Clara in Santa Clara, California, from June 23-August 29, 1980.

[8] T. Fukuda, S. Nakagawa, Y. Kawauchi, and M. Buss, "Self organizing robots based on cell structures—CEBOT". *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems* (IROS), Nov. 1988, pp. 145-150.

[9] S.K. Yun, D. Rus, "Self Assembly of Modular Manipulators with Active and Passive Modules", *Proc. of IEEE/RSJ IEEE International Conference on Robotics and Automation* , Pasadena, CA, May, 2008.

[10] F. Hou and W.M. Shen, "Mathematical Foundation for Hormone-Inspired Control for Self-Reconfigurable Robotic Systems", *Proc. 2006 IEEE Intl. Conf. on Robotics and Automation*, May, 2006.

[11] M. Yim, W.M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins and G.S. Chirickjian, "Modular Self-Reconfigurable Robot Systems", *IEEE Robotics and Automation Magazine*, Mar. 2007, Vol. 14, No. 1, pp. 43-42.

[12] B. Salemi, M. Moll, and W.M. Shen, "SUPERBOT: A Deployable, Multi-Functional, and Modular Self-Reconfigurable Robotic System", *Proc. 2006 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, October, 2006.

[13] E. Klavins, “Programmable Self-Assembly” *Control Systems Magazine,* Aug. 2007, Vol. 24, No. 4, pp. 43-56.

[14] F. Shaw and E. Klavins, “Distributed Estimation and Control For Stochastically Interacting Robots”. *IEEE Conference on Decision and Control*, 2008.

[15] E. Klavins, “Automatic Synthesis of Controllers for Distributed Assembly and Formation Forming”, *Proceedings of the IEEE Conference on Robotics and Automation*, May 2002.

[16] E. Klavins, R. Ghrist and D. Lipsky, ”A Grammatical Approach to Self-Organizing Robotic Systems”, *IEEE Transactions on Automatic Control*. Jun. 2006, Vol. 51, No. 5, pp. 949-962.

[17] M. Boncheva, D.A. Bruzewicz, and G.M. Whitesides, “Millimeter-Scale Self-Assembly and Its Applications”, *Pure Appl. Chem.* 75, 2003, pp. 621-630.

[18] N. Lynch, “Distributed Algorithms”, Morgan Kaufmann, 1996.

[19] H.P. Young, “Individual Strategy and Social Structure: An Evolutionary Theory of Institutions”, Princeton University Press, 1998, pp. 46.

[20] J. R. Marden and J. S. Shamma, “Revisiting Log-Linear Learning: Asynchrony, Completeness and a Payoff-based Implementation”, submitted to *Games and Economic Behavior*, 2008.

[21] N. Napp, S. Burden and E. Klavins, “Setpoint Regulation for Stochastically Interacting Robots”, *Robotics: Science and Systems V*, MIT Press, 2009.